# WHOLE BODY CONTROL OF WHEELED INVERTED PENDULUM HUMANOIDS

A Dissertation
Presented to
The Academic Faculty

By

Munzir Zafar

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology

August 2019

**WHOLE BODY CONTROL OF WHEELED INVERTED PENDULUM HUMANOIDS**

Approved by:

Dr. Seth Hutchinson, Advisor
School of Interactive Computing
*Georgia Institute of Technology*

Dr. Evangelos Theodorou
School of Aerospace Engineering
*Georgia Institute of Technology*

Dr. Christensen, Henrik I.
School of Electrical and Computer Engineering
*Georgia Institute of Technology*

Dr. Justin Romberg
School of Electrical and Computer Engineering
*Georgia Institute of Technology*

Dr. Byron Boots
School of Interactive Computing
*Georgia Institute of Technology*

Date Approved: April 24, 2019

If machines produce everything we need,

the outcome will depend on how things are distributed.

*Stephen Hawking*

Dedicated to Ammi

## ACKNOWLEDGEMENTS

This life is a test. Nothing taught me this better than my stint as a PhD student. I had it easy till I became a graduate student. So much so, that I had begun to take more interest in metaphysics because tech development seemed too easy to excite my passion. Only as a graduate student did I learn that science may be easy to grasp in the hindsight, but the discovery of scientific truths is hard, very hard, to come by. Though that is not the only test I had as a grad student. In my distraction with what seemed to me as more pressing life issues through my years in Georgia Tech as a PhD student—to do with many turbulent political events taking place in the part of the world where I come from—I did not pay attention to the crisis that had built up in my PhD life due to sudden passing away of professor Mike Stilman whose lab I had committed myself to for two years by then.

I realized pretty late in my PhD that I was dealt a trial worse than many, and I had to wake up to my immediate reality if I needed to survive the urge to give up, when Fulbright scholarship ended. I would have given up had it not been for my dear wife, Unaiza Ahsan, who also happened to be a PhD student in Tech and convinced me to continue. When I decided to tackle my problems head on, the realization matured that what I have at my hand is a very uphill task. I had stayed with the project of Golem Krang, even as Mike Stilman had passed away who had led its support and development till then, and even as all students had moved on, ending up with me being all on my own with this behemoth for a period of three years.

It is those who helped me through the period of three-and-a-half years that followed who I owe this PhD to. Starting with God—the Glorious and the Beneficent—Whom I became extremely close to and reliant on, and Who I fell in love with ever more deeply, as He helped me fight through my challenges without making me lose hope or compromise on my ethics. And then the unflinching confidence, support, and encouragement that was provided by my close ones—my parents, my dear wife Unaiza, and my childhood friend

Fazal Abbas.

For support in terms of backing, funding, and guidance, I begin by thanking Dr. Henrik Christensen, who had my back as I decided to defend my PhD proposal. Then I have to thank Dr. Evangelos Theodorou, without whose kindness and encouragement, as well as, the deeply intellectual environment he provided to me in his amazing lab, I would never have actualized my fullest potential as a researcher. I am then thankful to Dr. Byron Boots whose eventual interest in the robotic platform, Golem Krang, helped raise the interest of the Tech community in my research, and which finally earned me the interest and support of Dr. Seth Hutchinson, who I can't thank enough as he supported me in the final fifteen months of my PhD after joining Tech, as his amazing people skills attracted many helping hands to work with me on hardware experiments in the Summer of 2018, and as he offered insights in the final stages of my PhD, and who convinced me that I had done good enough work to be able to defend my thesis.

I can't thank enough the student collaborators who helped me: Beginning with Can Erdogan, who I learned everything about Golem Krang from in the first few months of joining Mike's lab. Then Areeb Mehmood, who joined me as a Masters student at a time when I was completely alone, and who stuck with me till the end through the last two years of my PhD. Then Akash Patel who worked with me in the last one year of my PhD, and whose enthusiasm and interest in the software design and development of Krang reminded me of Mike Stilman's spirit when it comes to software design. Then I have to thank Mouhyemen Khan who spent nights with me at critical work deadlines and offered invaluable support. I would like to thank all other student collaborators who offered help through the exciting Summer of 2018: Shimin Zhang, Victor Aladele, Ali Murtaza, Sergio Aguilera, Nathaniel Glaser, Vlahov Bogdan, Manisha Natarajan, Rishi Mishra, and Siddarth Agarwal.

Finally, I would like to acknowledge Fulbright and HEC Pakistan, who jointly funded my PhD in the first five years, the ECE department who funded me as GTA for three semesters, and Dr. Byron Boots who funded me in the final two semesters as a GRA.

## TABLE OF CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

## SUMMARY

We propose to develop a framework for controlling a *Wheeled Inverted Pendulum (WIP) Humanoid* to perform useful interactions with the environment while dynamically balancing itself on two wheels. As humanoid platforms are characterized by several degrees of freedom, they have the ability to perform several tasks simultaneously, such as balancing themselves, maintaining a specific body pose, controlling the gaze, lifting a load or carrying a tray of cups filled to the brim. These tasks are all performed simultaneously, and the whole body participates in achieving each objective, with priorities assigned to each. The control also has to operate within constraints of angle and torque limits on each joint, as well as safety constraints of avoiding self-collision and collision with obstacles. This problem is referred as *Whole-Body Control* in the wider humanoid literature, and several successful solutions have recently been demonstrated for bipedal humanoid platforms. Our focus in this work is to develop a framework for whole-body control of WIP humanoids that can be applied directly on the physical robot, which means that it can be made robust to modeling errors as well as able to incorporate constraints on control and state as mentioned above. The proposed approach is hierarchical with a low level controller responsible for controlling the manipulator/body and a high-level controller that defines center of mass (CoM) targets for the low-level controller to control zero dynamics of the system driving the wheels. The low-level controller plans for shorter horizons while considering more complete dynamics of the system, while the high-level controller plans for longer horizon based on an approximate model of the robot for computational efficiency. Our core contributions are

- Showing how to isolate the dynamics of the manipulator from those of the wheels such that the resulting model is amenable to existing whole-body control techniques, such as operational space control [1] and quadratic programs (QP) [2]
- Using differential dynamic programming (DDP) to generate optimal trajectories for center of mass to control wheel motion.

# CHAPTER 1
# INTRODUCTION

Wheeled inverted pendulum systems offer the best of two worlds. Their wheels make for inherently fast and efficient locomotion—something that bipedal system designers are still struggling to achieve. And the dynamically balancing inverted pendulum endows them with the ability to deal with very heavy payloads, their torques being canceled by a readily adjustable center of mass—something that statically stable wheeled platforms can not beat. These characteristics make them attractive for a wide range of applications such as Segway human transporters [3, 4, 5, 6, 7], transporters with seats [7, 8, 9, 6, 10], self-balancing wheel chairs [11, 12, 13, 14] and WIP Humanoids [15, 16, 17, 18, 19, 20, 21, 22, 23]. WIP Humanoids add to the abilities of a WIP by offering a redundant manipulator, with one or more arms, that can be controlled to intelligently interact with their environment and perform useful tasks. If they were controlled for safely handling large forces, their ability to dynamically cancel out their effects can prove useful for assisting humans with tasks requiring large effort. Golem Krang [20] was designed with this purpose in mind. Its two-arm-bearing torso is mounted on massive base and spine links, thus providing it with the ability to perform heavy tasks by manipulating its weight torque as needed.

The challenge in unlocking the enormous potential in these structures is their nonlinear, highly unstable and under-actuated dynamics. A large body of literature exists to deal with these typical problems of WIP systems. However, the focus of most studies remains a simplified system having one-link attached to the wheels. This has inspired some to leverage this work for WIP humanoid control, by treating the control of wheeled inverted pendulum independently from the control of upper body—the former being a simplified model of the full robot with one link of an equivalent center of mass (CoM). This technique was also utilized by our group in an earlier work [24] to control Golem Krang. The problems

associated with such a treatment become apparent when precision-critical or safety-critical tasks are performed. For example, since forward motion is left up to the wheels alone, and the upper body is not compensating for pitch changes induced by the demands of locomotion on the wheel controller, the end-effector can hardly obey constraints on its position and orientation during locomotion, which may sometimes be critical to perform a task. Similarly, when dealing with sudden changes in large forces, the transients induced on balancing controller due to the switching of equilibrium positions can not be properly managed if the body is blind to the control of its center of mass. This makes the case of a unified approach to locomotion and manipulation tasks. Concerning walking humanoids, a vast and rapidly growing literature exists to deal with this problem but similar work in the WIP domain is nearly non-existent. We propose to fill this gap in this thesis. Specifically, we propose the development of a framework that deals with whole-body control of WIP humanoid structures typified by the robot, Golem Krang, and we aim to demonstrate the application of the proposed solutions on this platform.

A brief outline of this document is as follows: Chapter 2 covers the related work. Chapter 3 describes the dynamic model of the system under consideration. Chapter 4 shows how the idea of isolating the manipulator/body dynamics from wheel dynamics will help leverage the existing techniques for walking humanoids. Chapter 5 deals with the control of wheel dynamics to achieve a desired ground navigation of the robot. Details of implementation and results are discussed in chapter 6. Estimation of CoM parameters of the robot is presented in chapter 7, followed by discussion of future work and conclusion in chapter 8.

# CHAPTER 2
# RELATED WORK

Here we analyze the relevance of existing techniques proposed for whole body control of humanoid structures (not necessarily WIP), for the control of general WIP structures (not necessarily humanoids) and techniques that specifically address the problem of whole-body control of WIP humanoids.

## 2.1 Whole-Body Control of Humanoids

Full body control of humanoids has attracted a lot of research interest recently. The approaches proposed are generally grounded in the broad framework of [25]. This framework was initially introduced for unconstrained fully-actuated serial robots with stationary platforms. It allows the specification of tasks in workspace coordinates directly, and the controller solves for the required joint trajectories autonomously. Tasks are specified as cartesian positions/orientation or forces/torques and thus have six or less dimensions. For redundant systems, *i.e.*, that have more degrees of freedom (DOFs) than individual task dimensions, the framework allows for multiple lower-dimensional tasks to be simultaneously performed. Convex optimization can be utilized within this framework to handle constraints and optimize costs.

This framework is extended for control of walking humanoids in [26, 27]. It is a hierarchical approach where several priorities are defined such that each lower-priority task is projected in the combined null-space of tasks with priorities higher to it. Constraints and objectives are handled using virtual linkages.

Other hierarchical approaches [28, 29, 30, 31, 32, 33, 34] explore the formulation of objectives and constraints as quadratic programs. Inverse dynamics (ID) approaches use system dynamics as one of the constraints when solving for target joint accelerations and

torques. [29] propose a generic solution to hierarchical quadratic programs that enforces equality and inequality constraints at all hierarchies, and is significantly faster than previous methods.

As opposed to hierarchical approaches that have hard priorities, [2] prefer using weighted cost terms to specify preferences among the objectives. The compromise on precision it leads to is a trade-off to avoid the problem of ill-conditioned matrix inversions prevalent in hierarchical null-space approaches. They directly optimize a quadratic cost in terms of state acceleration, torques and contact forces using the full robot model.

Using inverse kinematics (IK) for tracking joint positions is also popular for controlling humanoid robots. A damped least squares method is used by [35, 36]. Using a similar approach, [37] compute the generalized velocity by inverting a matrix composed of end effector and contact constraint Jacobian. Computing only the generalized speeds $\dot{q}_{ik}$ and then integrating it to compute the corresponding positions $q_{ik}$ converges to local minima, but it produces continuous results reactive to online changes in the desired motions since optimization is performed at every time step.

The framework we develop for WIP humanoid control is able to leverage this existing body of literature. We show this by isolating the manipulator/body dynamics from wheel dynamics. We present a detailed adaptation of the approach by [26] into this framework, but other approaches can also be easily adapted.

### 2.2 Control of Wheeled Inverted Pendulum Systems

Wheeled Inverted Pendulum systems have been subjected to various control techniques including linear techniques implemented on the linearized model [38, 39, 40, 8], partial feedback linearization [41], energy-shaping [42], fuzzy logic based control [43, 44], neural-network based control [45, 46, 5, 47], hybrid controllers that switch between multiple controllers in different modes [48, 49, 22] and evolutionary algorithms for optimizing controllers [50, 51, 52] among many others.

Most techniques in this body of work consider the problem of controlling a one-link wheeled inverted pendulum model where the wheel torque is the primary control input controlling both the forward motion of the wheels and the balance of the inverted pendulum. Several joint torques present in the WIP humanoid body make for many control inputs in our case. And so, balancing the system purely using wheel-torque is neither necessary nor preferable. The entire upper body should participate in carrying out body balance as well as locomotion. This type of formulation is considered only by [16, 17, 18] and will be critically analyzed in the next section. But it is worth noting here that we control the wheel motion indirectly by controlling the zero dynamics that result from upper body control. We propose to do this by precalculating an optimal control trajectory (discussed in Chapter 5) for upper body motions that minimize a tracking error for forward motion of the wheels. This allows the whole body to participate in controlling the center of mass as well as locomotion. The need to generate such a trajectory arises from the specific approach we take and therefore we rely on a novel controller based on differential dynamic programming (DDP), a technique suitable for finding globally optimal trajectories for nonlinear systems and can potentially be extended to incorporate adaptive control to deal with modeling errors.

### 2.3 Whole-Body Control of Wheeled Inverted Pendulum Humanoids

The specific problem of whole body control of WIP humanoids is addressed by Toshiyuki *et al*. [16, 17, 18]. The system under consideration in their work has a passive base joint. This means that the reaction torque due to wheel motors does not directly control the *base link*—a term we use to identify the first link of the body attached to the wheels. They use null-space control, at the kinematic level, on only the actuated DOFs to find their reference accelerations for joint-level controllers. The passive joint is indirectly stabilized to always remain at a fixed position. In [16], only the wheels are stabilizing the passive joint by considering a simplified dynamic model of the system—a two-link WIP with first link rep-

resenting the original passive joint and second representing the manipulator on top of it as a rigid body of an equivalent center of mass. In [18], all active joints participate in this stabilization. In this work, they also propose a prioritization scheme such that lower-priority tasks are projected in the null-space of the jacobians of higher-priority ones. They perform three tasks, in the following order of priority: passive joint stabilization, end-effector positioning and center of mass control. The goal position of end-effector is in workspace coordinates in an inertial frame fixed on the ground, and their controller naturally generates locomotion trajectory if the goal location is not within immediate reach of the robot.

The problem of whole-body control of WIP humanoids with an active base joint however has not been addressed in the existing body of literature. In this case, the base joint is directly actuated by the reaction torque of the wheel motors. This happens when the wheel motors are mounted directly on the base link. This class of WIP systems are easier to build mechanically but pose a more challenging control problem due to the dynamics of the base link and the wheels being coupled by the wheel torque, in addition to the usual inertial coupling normally experienced by robotic systems. Most of the literature on WIP systems focuses on the first class of under-actuation (*i.e.*, passive base joint), which is mechanically more complex to build but offers a relatively easier control as the additional actuator torque coupling is not present. This fact is also noted by [53].

Due to this difference, our approach fundamentally differs from Toshiyuki *et al*. as their strategy treats the motion of the passive joint as the zero dynamics they indirectly control through actuated joint motions. Also, the base link in the structure they consider is a "platform" on top of which the fully actuated structure is mounted. They restrain this platform to stay at a fixed angular position. In general however, this is not necessary, as restricting the base link hinders its participation with other joints in performing the desired control tasks. The system we consider treats the base link as part of the manipulator/body performing the task, and the zero dynamics of the system are then derived that result from that control. The tasks for upper body are specified in a local frame of reference, which is

fixed on the base such that it is stationary with respect to the upper body but moves with robot locomotion. Wheel motion is then controlled indirectly by finding optimal trajectories of body links to affect the zero dynamics to achieve that end. This strategy, as we will see, can easily accommodate the application of wide range of techniques offered by existing literature on redundant manipulators fixed on stationary platforms, once the issues that result from the wheel torque coupling are resolved. Robot locomotion is then treated as a higher-level task planned according to the desired workspace objectives.

Also, we demonstrate the application of inverse dynamics based approaches on WIP systems, as opposed to purely kinematics based approach proposed by Toshiyuki *et al*. Approaches based on inverse dynamics offer robustness to disturbances such as external force changes and result in compliant systems. However, only using inverse dynamics can perform poorly due to inevitable modeling errors generally unavoidable on real systems. So for implementation it is still important to generate kinematic targets for accurate motion.

# CHAPTER 3
# DYNAMIC MODEL

## 3.1 Introduction

In this chapter, we present the dynamic model of a Wheeled Inverted Pendulum humanoid—Golem Krang. The techniques usually utilized for dynamic modeling of a WIP system are Euler-Lagrange, Newton's laws of motion or Kane's method. Assumptions include rigid bodies, flat ground and zero slip. Another simplification for dynamic modeling is to ignore the third dimension, resulting in a planar robot (meaning it does not have yaw). Such models are helpful because it is not the yaw but the pitch of the WIP system that forms the complexity of under-actuation as it shares its actuator with the wheels, so control of planar robots can be extended to include 3D robots. Work that derive dynamics of planar case include [54, 55, 56, 57, 39, 58, 59, 60]. Our analysis does not ignore the third dimension, and our robot is allowed to explore the full range of accessible workspace. Among those who have considered the yaw are [38, 61, 62, 63, 64]. However, they perform their analysis on simple WIP systems, with the body consisting of either just one link, or a simplifying assumption is used to treat it as such. Our analysis does not make any such simplifying assumptions in the derivation of the model. We make the simplifying assumption of assuming a planar robot in control design (Chapters 4 and 5), but we plan to extend those techniques to full 3D model derived here as part of our future work.

The system under consideration is a tree-structured serial robot, with two 7-DOF arms branching out from a 3-DOF serial structure mounted on a differential wheel drive. We derive the full model of the system using Kane's formulation, followed by explicit expressions for terms that appear in its zero dynamics. We also go over the steps that were skipped in deriving the simplified model. Also contained here is an algorithmic summary of the pro-

posed hierarchical approach. Finally, we present the table of values used for simulation and control parameters in the results section of the original paper. This material is organized as follows: Section 3.2 introduces the frames of references and the generalized coordinates. Section 3.3 introduces the Kane's formulation. Section 3.4 derives the expressions for the left-hand side of the equations of motions. Section 3.5 derives the terms on the right-hand side. Section 3.6 evaluates detailed expressions for the zero dynamics of the system. Section 3.7 derives using our full model, the dynamic model of a 2D WIP system. Section 3.8 gives the concluding remarks.

### 3.2 Generalized Coordinates

Figure 3.1 shows the inertial frame $\Sigma$ fixed on ground (with axes $x$, $y$ and $z$ and unit vectors $\bar{I}$, $\bar{J}$ and $\bar{K}$) and the frames of reference $\mathbb{F} = \{0, 1, 2, 3, 4l, ... , 10l, 4r, ... , 10r\}$. Each frame $j \in \mathbb{F}$ is fixed on a link of the tree structure, with the exception frame $0$, the base frame, which locates the robot in the world frame ($\Sigma$), specifically the mid-point between its two wheels. For the derivation, we define $\mathbb{F}_j \subset \mathbb{F}$ as the set of all frames succeeding frame $j$ in the tree-structure *e.g.* $\mathbb{F}_2 = \{3, 4l, ... , 10l, 4r, ... , 10r\}$, $\mathbb{F}_{5l} = \{6l, ... , 10l\}$ etc. and $\mathbb{G}_j \subset \mathbb{F}$ as the set of all frames antecedent to frame $j$ *e.g.* $\mathbb{G}_3 = \{0, 1, 2\}$, $\mathbb{G}_{7r} = \{0, 1, 2, 3, 4r, 5r, 6r\}$ etc.

Each frame $j$ moves at angular velocity $\dot{q}_j \bar{e}_j$ relative to its preceding link, where $\dot{q}_j$ and $\bar{e}_j$ are respectively its magnitude and direction. Origin of frame $j$ is denoted by $O_j$, axes by $x_j$, $y_j$ and $z_j$ and the respective unit vectors by $\bar{i}_j$, $\bar{j}_j$ and $\bar{k}_j$. Then $O_0$ is at the mid-point between the wheels, $\bar{i}_0$ is the heading direction of the robot and $\bar{k}_0$ is always perpendicular to the ground. Apart from the link rotations, six more coordinates will be defined to represent the configuration of the base frame and wheels.

$\theta_L, \theta_R$ are the rotation angles of the left and right wheels respectively

$X_0, Y_0$ are the position coordinates of the base frame in $\Sigma$

$\psi$ is the angle denoting heading direction (*i.e.*, angle between $\bar{i}_0$ and $x$-axis of $\Sigma$). Thus $\dot{\psi}$ represents the spin speed of the base frame.

$\dot{x}$ is the heading speed of the base frame (*i.e.*, along $\bar{i}_0$)

The full set of generalized co-ordinates is

$$\mathbf{q}_{full} = \begin{bmatrix} X_0 & Y_0 & \theta_L & \theta_R & q_1 & q_2 & q_3 & q_{4l} & \dots & q_{10l} & q_{4r} & \dots & q_{10r} \end{bmatrix}^\top$$

Finally Angular velocity of frame 0 is $\dot{\psi}\bar{k}_0$. So we define $\dot{q}_0 = \dot{\psi}$ and $\bar{e}_0 = \bar{k}_0$. This will help with closed form expressions of tree-structure kinematics and dynamics. In the derivation that follows, $\bar{v}_A$, $\bar{\omega}_A$, $\bar{a}_A$ and $\bar{\alpha}_A$ denote respectively the linear velocity, angular velocity, linear acceleration and angular acceleration of point $A$. Also, $\bar{r}_{A/B}$ denotes the position of point $A$ with respect to $B$.

### 3.2.1 Non-holonomic Constraints

Under the assumption of no slipping/skidding, we have two constraint equations:

$$\bar{v}_L = R\dot{\theta}_L\bar{i}_0 \qquad \bar{v}_R = R\dot{\theta}_R\bar{i}_0 \tag{3.1}$$

Since

$$\begin{aligned}
\bar{v}_L &= \bar{v}_0 + \bar{\omega}_0 \times \bar{r}_{L/O} \\
&= \dot{X}_0\bar{I} + \dot{Y}_0\bar{J} + \dot{\psi}\bar{K} \times (\frac{L}{2}\bar{j}_0) \\
&= \dot{X}_0(\cos\psi\bar{i}_0 - \sin\psi\bar{j}_0) + \dot{Y}_0(\sin\psi\bar{i}_0 + \cos\psi\bar{k}_0) - \frac{L}{2}\dot{\psi}\bar{i}_0 \\
&= (\dot{X}_0\cos\psi + \dot{Y}_0\sin\psi - \frac{L}{2}\dot{\psi})\bar{i}_0 - (\dot{X}_0\sin\psi - \dot{Y}_0\cos\psi)\bar{j}_0
\end{aligned}$$

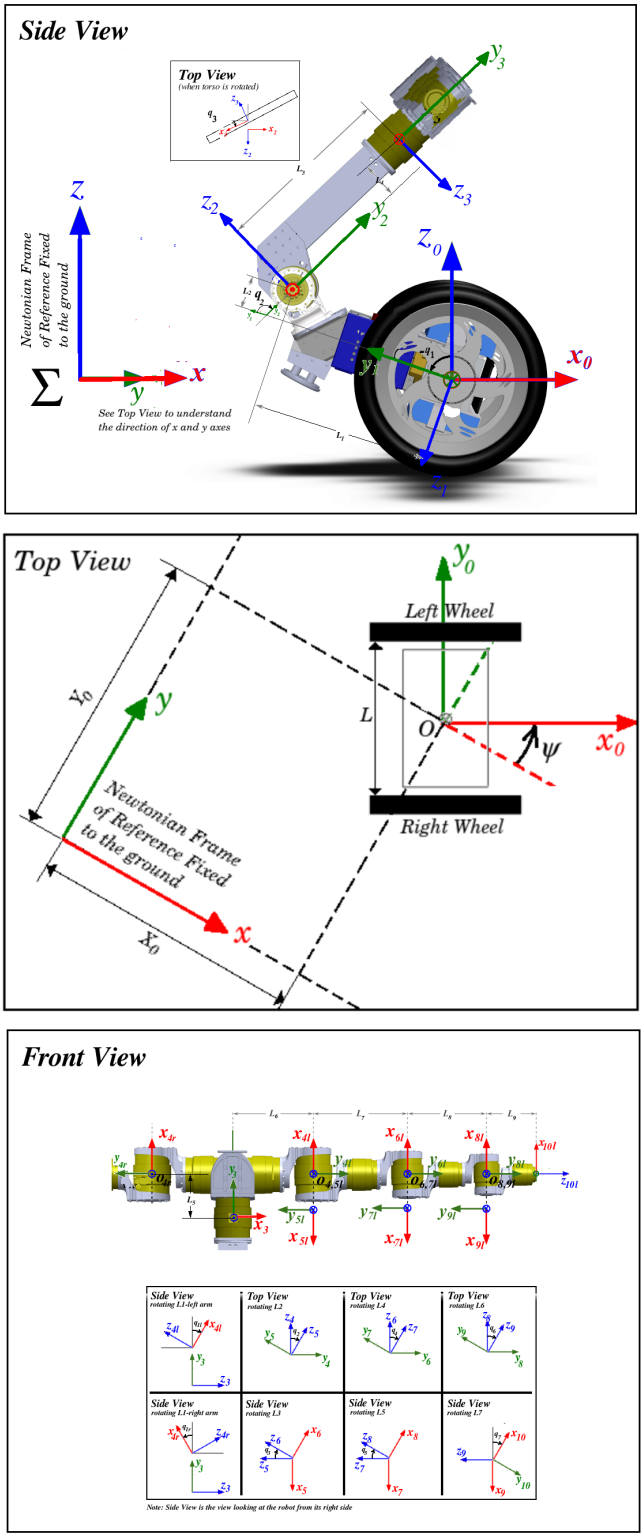



Figure 3.1: Frames of references on the robot

And similarly,

$$\bar{v}_R = (\dot{X}_0 \cos\psi + \dot{Y}_0 \sin\psi + \frac{L}{2}\dot{\psi})\bar{i}_0 - (\dot{X}_0 \sin\psi - \dot{Y}_0 \cos\psi)\bar{j}_0$$

Comparing the coefficients of $\bar{i}_0$ in (3.1) gives:

$$\begin{aligned} \dot{X}_0 \cos\psi + \dot{Y}_0 \sin\psi - \frac{L}{2}\dot{\psi} &= R\dot{\theta}_L \\ \dot{X}_0 \cos\psi + \dot{Y}_0 \sin\psi + \frac{L}{2}\dot{\psi} &= R\dot{\theta}_R \end{aligned} \tag{3.2}$$

Subtraction and addition of the two equations gives us:

$$\dot{\psi} = \frac{R}{L}(\dot{\theta}_R - \dot{\theta}_L) \tag{3.3}$$

$$\dot{X}_0 \cos\psi + \dot{Y}_0 \sin\psi = \frac{R}{2}(\dot{\theta}_L + \dot{\theta}_R) \tag{3.4}$$

Integrating (3.3) gives $\psi = \frac{R}{L}(\theta_R - \theta_L)$ which can be substituted in (3.4) to give the first constraint equation relating the first four generalized co-ordinates.

Comparing the coefficients of $\bar{j}_0$ in (3.1) gives the second constraint equation

$$\dot{X}_0 \sin\psi - \dot{Y}_0 \cos\psi = 0 \tag{3.5}$$

Twenty-one generalized coordinates, with two constraints (3.4)-(3.5) leads to nineteen degrees of freedom.

### 3.2.2 Defining Generalized Velocities

It is easier to derive the dynamic model of the system in terms of the generalized velocities:

$$\dot{\mathbf{q}} = \begin{bmatrix} \dot{x} & \dot{\psi} & \dot{q}_1 & \dot{q}_2 & \dot{q}_3 & \dot{q}_{4l} & \dots & \dot{q}_{10l} & \dot{q}_{4r} & \dots & \dot{q}_{10r} \end{bmatrix}^\top \tag{3.6}$$

To avoid confusion let us emphasize that the non-bold reference to generalized speeds $q_j$, $j \in \mathbb{F}$ refers to the rotation speed of frame $j$. However, the bold-faced notation $\dot{\mathbf{q}}_j$, $j \in \{1, ..., 19\}$ refers to elements of the vector in (3.6). These nineteen velocities can take arbitrary values which are all kinematically admissible. In other words, they represent our nineteen degrees of freedom. Here $\dot{x}$ should be refered to as a quasi-velocity as this velocity has meaning only as a velocity but its corresponding position variable $x$ does not give any physical meaning. Although, the infinitesimal change of position $\delta x = \dot{x}dt$ (sometimes refered to as virtual displacement) has physical meaning.

Full set of speeds can be calculated using

$$\dot{X}_0 = \dot{x}\cos\psi \qquad \dot{Y}_0 = \dot{x}\sin\psi \qquad \dot{\theta}_L = \frac{1}{R}\dot{x} - \frac{L}{2R}\dot{\psi} \qquad \dot{\theta}_R = \frac{1}{R}\dot{x} + \frac{L}{2R}\dot{\psi} \tag{3.7}$$

Where the first two relationships are derived by comparing the coefficients in $\dot{X}_0\bar{I} + \dot{Y}_0\bar{J} = \bar{v}_0 = \dot{x}\bar{i}_0 = \dot{x}(\cos\psi\bar{I} + \sin\psi\bar{J})$. And the next two relationships are derived by substituting the the first two relationships in (3.2). When these relationships 3.7 are substituted in our constraint equations 3.4-3.5, both sides of the equations vanish, indicating that these nineteen degrees of freedom are not bound by any constraint. We will now derive nineteen dynamic equations in terms of our new generalized velocities. Those equations in conjunction with these relationships can solve for all twenty-one generalized coordinates.

We will be using the Kane's formulation. This is done because the presence of quasi-velocity prohibits us from using Lagrange method for dynamic modeling.

### 3.3 Introduction to Kane's Formulation

$$\sum_k \left[ m_k \bar{a}_{Gk} \cdot (\bar{v}_{Gk})_j + \left( \frac{d\bar{H}_{Gk}}{dt} \right) \cdot (\bar{\omega}_k)_j \right] \tag{3.8}$$
$$= \sum_n \bar{F}_n \cdot (\bar{v}_n)_j + \sum_n \bar{M}_m \cdot (\bar{\omega}_m)_j \quad j = 1...19$$

where

$j$ corresponds to the elements of the generalized speed vector $\dot{\mathbf{q}}$ defined in (3.6)

$k$ corresponds to rigid bodies in the system

$n$ corresponds to external forces acting on the system

$m$ corresponds to external torques acting on the system

$m_k$ mass of the body $k$

$\bar{a}_{Gk}$ CoM acceleration of body $k$

$\bar{v}_{Gk}$ CoM velocity of body $k$

$\bar{H}_{Gk}$ angular momentum of body $k$ about its CoM

$\bar{\omega}_k$ angular velocity of body $k$

$\bar{F}_n$ $n$th external force

$\bar{M}_m$ $m$th external moment

$\bar{v}_n$ velocity of the point at which external Force $\bar{F}_n$ is acting

$\bar{\omega}_m$ angular velocity of the body on which torque $\bar{M}_m$ is acting

$()_j = \frac{\partial()}{\partial \dot{\mathbf{q}}_j}$ partial derivative of the quantity in brackets $()$ with respect to the generalized speed $\dot{\mathbf{q}}_j$

### 3.4 Kane's Left Hand Side

The left hand side of the Kane's equation contains a sum whose range is equal to the number of bodies in the system. We have nineteen bodies, refered by $k \in \{L, R\} \cup \mathbb{F}_0$. These include left-wheel ($L$), right wheel ($R$) and seventeen links in the tree structure of the

robot, which we denote by their respective frames $j \in \mathbb{F}_0$. Each term in the sum consists of the acceleration ($\bar{a}_{Gk}$), velocity ($\bar{v}_{Gk}$), angular momentum ($\bar{H}_{Gk}$) of the center of mass and the body's angular velocity ($\bar{\omega}_k$). And then some partial derivatives wrt to the generalized velocities ($(\bar{\omega}_k)_j = \frac{\partial \bar{\omega}_k}{\partial \dot{\mathbf{q}}_j}$ and $(\bar{v}_{Gk})_j = \frac{\partial \bar{v}_{Gk}}{\partial \dot{\mathbf{q}}_j}$).

### 3.4.1 Frame 0

The velocities and accelerations of the frame 0 are needed to evaluate the afore-mentioned quantities related to each body. These velocities are as follows:

$$\begin{aligned} \bar{\omega}_0 &= \dot{\psi}\bar{k}_0 \qquad \bar{v}_0 = \dot{x}\bar{i}_0 \qquad \bar{\alpha}_0 = \ddot{\psi}\bar{k}_0 \\ \bar{a}_0 &= \ddot{x}\bar{i}_0 + \dot{x}\left(\dot{\psi}\bar{k}_0 \times \bar{i}_0\right) = \ddot{x}\bar{i}_0 + \dot{x}\dot{\psi}\bar{j}_0 \end{aligned} \tag{3.9}$$

### 3.4.2 Wheels

*Left Wheel*

This evaluation takes place in the $x_L y_L z_L$ frame fixed to the left wheel such that it is parallel to frame $x_0 y_0 z_0$ when $\theta_L = 0$. So $\bar{i}_0 = \cos\theta_L \bar{i}_L + \sin\theta_L \bar{k}_L$, $\bar{j}_0 = \bar{j}_L$ and $\bar{k}_0 = -\sin\theta_L \bar{i}_L + \cos\theta_L \bar{k}_L$. Velocities:

$$\begin{aligned} \bar{\omega}_L &= \dot{\psi}\bar{k}_0 + \dot{\theta}_L \bar{j}_0 \\ &= \dot{\psi}\bar{k}_0 + \left(\tfrac{1}{R}\dot{x} - \tfrac{L}{2R}\dot{\psi}\right)\bar{j}_0 \\ &= -\dot{\psi}\sin\theta_L \bar{i}_L + \dot{\psi}\cos\theta_L \bar{k}_L \\ &\qquad + \left(\tfrac{1}{R}\dot{x} - \tfrac{L}{2R}\dot{\psi}\right)\bar{j}_L \end{aligned}$$

$$\begin{aligned}\bar{v}_{GL} &= \bar{v}_0 + \bar{\omega}_0 \times \bar{r}_{L/O} \\ &= \dot{x}\bar{i}_0 + \dot{\psi}\bar{k}_0 \times \tfrac{L}{2}\bar{j}_0 \\ &= \left(\dot{x} - \tfrac{L}{2}\dot{\psi}\right)\bar{i}_0 \end{aligned} \tag{3.10}$$

Linear acceleration, angular momentum and its derivative:

$$\begin{aligned}\bar{a}_{GL} &= \bar{a}_0 + \bar{\alpha}_0 \times \bar{r}_{L/O} + \bar{\omega}_0 \times \left(\omega_0 \times \bar{r}_{L/O}\right) \\ &= \ddot{x}\bar{i}_0 + \dot{x}\left(\dot{\psi}\bar{k}_0 \times \bar{i}_0\right) + \ddot{\psi}\bar{k}_0 \times \frac{L}{2}\bar{j}_0 \\ &\quad + \dot{\psi}\bar{k}_0 \times \left(\dot{\psi}\bar{k}_0 \times \frac{L}{2}\bar{j}_0\right) \\ &= \left(\ddot{x} - \frac{L}{2}\ddot{\psi}\right)\bar{i}_0 + \left(\dot{x}\dot{\psi} - \frac{L}{2}\dot{\psi}^2\right)\bar{j}_0 \end{aligned}$$

$$\begin{aligned}\bar{H}_{GL} &= I_w \bar{\omega}_L \\ \frac{d\bar{H}_{GL}}{dt} &= \frac{\partial \bar{H}_{GL}}{\partial t} + \bar{\omega}_L \times \bar{H}_{GL}\end{aligned} \tag{3.11}$$

where $I_w = \begin{bmatrix} \mathbf{ZZ}_w & 0 & 0 \\ 0 & \mathbf{YY}_w & 0 \\ 0 & 0 & \mathbf{ZZ}_w \end{bmatrix}$. Due to symmetry the off-diogonal terms in the inertia matrix vanish, and the inertia about $x_L$-axis and $z_L$-axis are both equal (signified by $\mathbf{ZZ}_w$).

*Right Wheel*

This evaluation takes place in the $x_R y_R z_R$ frame fixed to the right wheel such that it is parallel to frame $x_0 y_0 z_0$ when $\theta_R = 0$. So $\bar{i}_0 = \cos\theta_R \bar{i}_R + \sin\theta_R \bar{k}_R$, $\bar{j}_0 = \bar{j}_R$ and $\bar{k}_0 =$

$-\sin\theta_R\bar{i}_R + \cos\theta_R\bar{k}_R$. Angular velocity:

$$\begin{aligned}
\bar{\omega}_R &= \dot{\psi}\bar{k}_0 + \dot{\theta}_R\bar{j}_0 \\
&= \dot{\psi}\bar{k}_0 + \left(\tfrac{1}{R}\dot{x} + \tfrac{L}{2R}\dot{\psi}\right)\bar{j}_0 \\
&= -\dot{\psi}\sin\theta_L\bar{i}_R + \dot{\psi}\cos\theta_L\bar{k}_R \\
&\qquad + \left(\tfrac{1}{R}\dot{x} + \tfrac{L}{2R}\dot{\psi}\right)\bar{j}_R
\end{aligned}$$

$$\begin{aligned}
\bar{v}_{GR} &= \bar{v}_0 + \bar{\omega}_0 \times \bar{r}_{R/O} \\
&= \dot{x}\bar{i}_0 + \dot{\psi}\bar{k}_0 \times \left(-\tfrac{L}{2}\bar{j}_0\right) \\
&= \left(\dot{x} + \tfrac{L}{2}\dot{\psi}\right)\bar{i}_0
\end{aligned} \tag{3.12}$$

Linear acceleration, angular momentum and its derivative:

$$\begin{aligned}
\bar{a}_{GR} &= \bar{a}_0 + \bar{\alpha}_0 \times \bar{r}_{R/O} + \bar{\omega}_0 \times \left(\omega_0 \times \bar{r}_{R/O}\right) \\
&= \ddot{x}\bar{i}_0 + \dot{x}\left(\dot{\psi}\bar{k}_0 \times \bar{i}_0\right) + \ddot{\psi}\bar{k}_0 \times \left(-\frac{L}{2}\bar{j}_0\right) \\
&\quad + \dot{\psi}\bar{k}_0 \times \left(\dot{\psi}\bar{k}_0 \times \left(-\frac{L}{2}\bar{j}_0\right)\right) \\
&= \left(\ddot{x} + \frac{L}{2}\ddot{\psi}\right)\bar{i}_0 + \left(\dot{x}\dot{\psi} + \frac{L}{2}\dot{\psi}^2\right)\bar{j}_0
\end{aligned}$$

$$\begin{aligned}
\bar{H}_{GR} &= I_w\bar{\omega}_R \\
\frac{d\bar{H}_{GR}}{dt} &= \frac{\partial\bar{H}_{GR}}{\partial t} + \bar{\omega}_R \times \bar{H}_{GR}
\end{aligned} \tag{3.13}$$

*Wheel Contribution to Kane's LHS*

The contribution of wheels to LHS of the equation corresponding to generalized speed $\dot{\mathbf{q}}_j$ is:

$$\left(m_w \bar{a}_{GL} \cdot \frac{\partial \bar{v}_{GL}}{\partial \dot{\mathbf{q}}_j} + \left(\frac{d\bar{H}_{GL}}{dt}\right) \cdot \frac{\partial \bar{\omega}_L}{\partial \dot{\mathbf{q}}_j}\right) + \left(m_w \bar{a}_{GR} \cdot \frac{\partial \bar{v}_{GR}}{\partial \dot{\mathbf{q}}_j} + \left(\frac{d\bar{H}_{GR}}{dt}\right) \cdot \frac{\partial \bar{\omega}_R}{\partial \dot{\mathbf{q}}_j}\right) \tag{3.14}$$

Since:

| $\dot{\mathbf{q}}_j$ : | $\dot{x}$ | $\dot{\psi}$ | others |
|---|---|---|---|
| $\frac{\partial \bar{v}_{GL}}{\partial \dot{\mathbf{q}}_j}$ : | $\cos\theta_L \bar{i}_L + \sin\theta_L \bar{k}_L$ | $-\frac{L}{2}\left(\cos\theta_L \bar{i}_L + \sin\theta_L \bar{k}_L\right)$ | 0 |
| $\frac{\partial \bar{v}_{GR}}{\partial \dot{\mathbf{q}}_j}$ : | $\cos\theta_R \bar{i}_R + \sin\theta_R \bar{k}_R$ | $\frac{L}{2}\left(\cos\theta_R \bar{i}_R + \sin\theta_R \bar{k}_R\right)$ | 0 |
| $\frac{\partial \bar{\omega}_L}{\partial \dot{\mathbf{q}}_j}$ : | $\frac{1}{R}\bar{j}_L$ | $-\sin\theta_L \bar{i}_L - \frac{L}{2R}\bar{j}_L + \cos\theta_L \bar{k}_L$ | 0 |
| $\frac{\partial \bar{\omega}_R}{\partial \dot{\mathbf{q}}_j}$ : | $\frac{1}{R}\bar{j}_R$ | $-\sin\theta_R \bar{i}_R + \frac{L}{2R}\bar{j}_R + \cos\theta_R \bar{k}_R$ | 0 |

(3.15)

Therefore the contribution is only there for the first two equations (*i.e.*, for $\dot{\mathbf{q}}_j \in \{\dot{x}, \dot{\psi}\}$) and is zero for all other equations. The final expressions are listed in (3.16), found by substituting values from (3.10)-(3.13) and (3.15) in (3.14)

$$\begin{aligned} \dot{x} : &\quad \left(2m_w + \tfrac{2\mathbf{YY}_w}{R^2}\right)\ddot{x} \\ \dot{\psi} : &\quad \left(\tfrac{m_w L^2}{2} + \tfrac{\mathbf{YY}_w L^2}{2R^2} + 2\mathbf{ZZ}_w\right)\ddot{\psi} \end{aligned} \tag{3.16}$$

### 3.4.3 Serial Tree-Structure

*Recusrive Formulae*

The angular and linear velocities of the frames on the rest of the robot can be calculated using the recursive formulation common in serial robot literature and briefly introduced

here. Kinematic relationships for each link $j$ with immediately antecedent link $j^-$ are

$$\begin{aligned}
\bar{\omega}_j &= \bar{\omega}_{j^-} + \dot{q}_j\, \bar{e}_j \\
\bar{\alpha}_j &= \bar{\alpha}_{j^-} + \ddot{q}_j\, \bar{e}_j + \dot{q}_j\, (\bar{\omega}_j \times \bar{e}_j) \\
\bar{v}_j &= \bar{v}_{j^-} + \bar{\omega}_{j^-} \times \bar{r}_{O_j/O_{j^-}} \\
\bar{a}_j &= \bar{a}_{j^-} + \bar{\alpha}_{j^-} \times \bar{r}_{O_j/O_{j^-}} + \bar{\omega}_{j^-} \times (\bar{\omega}_{j^-} \times \bar{r}_{O_j/O_{j^-}})
\end{aligned} \tag{3.17}$$

where $\bar{\omega}_j$, $\bar{\alpha}_j$, $\bar{v}_j$ and $\bar{a}_j$ are respectively the angular velocity, angular acceleration, linear velocity and linear acceleration of frame $j$ relative to the Newtonian frame of reference (or the world frame). To make algebraic manipulations more explicit we note that, in practice, each vector in (3.17) is represented by three elements representing its projection on the axes of a particular reference frame. Each equation in (3.17) is therefore three scalar equations. Denote ${}^i\omega_j$ as the 3-element vector representing $\bar{\omega}_j$ in frame $i$. Then the recursive formulae will involve transformations of the vector representations described in (3.18)

$$\begin{aligned}
{}^j\omega_j &= {}^jA_{j^-}\, {}^j\bar{\omega}_{j^-} + \dot{q}_j\, {}^je_j \\
{}^j\alpha_j &= {}^jA_{j^-}\, {}^j\bar{\alpha}_{j^-} + \ddot{q}_j\, {}^je_j + \dot{q}_j\, ({}^j\omega_j \times {}^je_j) \\
{}^jv_j &= {}^jA_{j^-} \left( {}^j\bar{v}_{j^-} + {}^j\bar{\omega}_{j^-} \times {}^j\bar{P}_j \right) \\
{}^ja_j &= {}^jA_{j^-} \left( {}^j\bar{a}_{j^-} + {}^j\bar{\alpha}_{j^-} \times {}^j\bar{P}_j + {}^j\bar{\omega}_{j^-} \times ({}^j\bar{\omega}_{j^-} \times {}^j\bar{P}_j) \right)
\end{aligned} \tag{3.18}$$

where

- ${}^jA_i$ Rotation operator that transforms vector represented in frame $i$ to the same vector represented in frame $j$
- ${}^iP_j$ Position vector $\bar{r}_{O_j/O_i}$ represented in frame $i$

*Transformations*

Homogenous transformation of frame $j$ to frame $i$

$$ {}^{i}T_j = \begin{bmatrix} {}^{i}s_j & {}^{i}n_j & {}^{i}a_j & {}^{i}P_j \end{bmatrix} = \begin{bmatrix} {}^{i}A_j & {}^{i}P_j \\ 0_{3\times 1} & 1 \end{bmatrix} \tag{3.19} $$

where ${}^{i}s_j$, ${}^{i}n_j$ and ${}^{i}a_j$ contain the components of the unit vectors along the $x_j$, $y_j$ and $z_j$ axes respectively expressed in frame $i$, and where ${}^{i}P_j$ is the vector representing the coordinates of the origin of frame $j$ expressed in frame $i$.

Using figure 3.1, we can write down these transformation matrices for our system as follows:

$$
\begin{gathered}
{}^{0}T_1 = \begin{bmatrix} 0 & \mathrm{s}q_1 & -\mathrm{c}q_1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & \mathrm{c}q_1 & \mathrm{s}q_1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^{1}T_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \mathrm{c}q_2 & \mathrm{s}q_2 & L_1 \\ 0 & -\mathrm{s}q_2 & \mathrm{c}q_2 & -L_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^{2}T_3 = \begin{bmatrix} -\mathrm{c}q_3 & 0 & \mathrm{s}q_3 & 0 \\ 0 & 1 & 0 & L_3 \\ -\mathrm{s}q_3 & 0 & -\mathrm{c}q_3 & L_4 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\
{}^{3}T_{4l} = \begin{bmatrix} 0 & 1 & 0 & L_6 \\ \mathrm{c}q_{4l} & 0 & -\mathrm{s}q_{4l} & L_5 \\ -\mathrm{s}q_{4l} & 0 & -\mathrm{c}q_{4l} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^{3}T_{4r} = \begin{bmatrix} 0 & -1 & 0 & -L_6 \\ \mathrm{c}q_{4r} & 0 & -\mathrm{s}q_{4r} & L_5 \\ \mathrm{s}q_{4r} & 0 & \mathrm{c}q_{4r} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^{4a}T_{5a} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -\mathrm{c}q_{5a} & -\mathrm{s}q_{5a} & 0 \\ 0 & -\mathrm{s}q_{5a} & \mathrm{c}q_{5a} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\
{}^{5a}T_{6a} = \begin{bmatrix} -\mathrm{c}q_{6a} & 0 & \mathrm{s}q_{6a} & 0 \\ 0 & -1 & 0 & -L_7 \\ \mathrm{s}q_{6a} & 0 & \mathrm{c}q_{6a} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^{6a}T_{7a} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -\mathrm{c}q_{7a} & -\mathrm{s}q_{7a} & 0 \\ 0 & -\mathrm{s}q_{7a} & \mathrm{c}q_{7a} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^{7a}T_{8a} = \begin{bmatrix} -\mathrm{c}q_{8a} & 0 & \mathrm{s}q_{8a} & 0 \\ 0 & -1 & 0 & -L_8 \\ \mathrm{s}q_{8a} & 0 & \mathrm{c}q_{8a} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \\
{}^{8a}T_{9a} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & -\mathrm{c}q_{9a} & -\mathrm{s}q_{9a} & 0 \\ 0 & -\mathrm{s}q_{9a} & \mathrm{c}q_{9a} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, {}^{9a}T_{10a} = \begin{bmatrix} -\mathrm{c}q_{10a} & -\mathrm{s}q_{10a} & 0 & 0 \\ 0 & 0 & -1 & -L_9 \\ \mathrm{s}q_{10a} & -\mathrm{c}q_{10a} & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \begin{array}{l} \text{where} \\ a \in \{l, r\}, \\ \mathrm{c}q_j = \cos q_j, \\ \mathrm{s}q_j = \sin q_j \end{array}
\end{gathered}
\tag{3.20}
$$

These transformations provide ${}^{j}T_j$. In (3.18), ${}^{j}A_{j^-}$ is needed and can be deduced from the identity

$$ {}^{j}A_i = {}^{i}A_j^{\top} \tag{3.21} $$

*Local Angular Velocity Directions*

The unit vectors along the direction of angular velocities of the frames in the tree-structure each represented in its local frame are as follows (using figure 3.1).

$$
\begin{aligned}
&{}^{0}e_0 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^\top, {}^{1}e_1 = \begin{bmatrix} -1 & 0 & 0 \end{bmatrix}^\top, {}^{2}e_2 = \begin{bmatrix} -1 & 0 & 0 \end{bmatrix}^\top, {}^{3}e_3 = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^\top, \\
&{}^{4a}e_{4a} = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^\top, {}^{5a}e_{5a} = \begin{bmatrix} -1 & 0 & 0 \end{bmatrix}^\top, {}^{6a}e_{6a} = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^\top, {}^{7a}e_{7a} = \begin{bmatrix} -1 & 0 & 0 \end{bmatrix}^\top, \\
&{}^{8a}e_{8a} = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^\top, {}^{9a}e_{9a} = \begin{bmatrix} -1 & 0 & 0 \end{bmatrix}^\top, {}^{10a}e_{10a} = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^\top, a \in \{l, r\}
\end{aligned} \tag{3.22}
$$

The information provided in (3.9), (3.19)-(3.22) can now be used in the recursive formulation (3.18) to derive expressions for the velocities and accelerations of the frames, using any symbolic language toolbox. To derive full expression of each quantity in terms of generalized co-ordinates is inefficient. In [65], Kane *et al.* have presented how the Kane's dynamical formulation can provide an efficient mechanism to evaluate expressions for serial robotic structures. It involves the use of intermediate variables $Z_i$ that encapsulate the result of evaluation at each recursion step, in both "implicit" and "explicit" forms. Explicit form only encapsulates the coefficients of generalized speeds and accelerations ($\dot{q}_i$, $\ddot{q}_i$), *e.g.* ${}^{2}\omega_2 = Z_9\dot{x} + Z_{10}\dot{q}_1 + Z_{11}\dot{q}_2$, as opposed to the implicit form where an intermediate variable will encapsulate the whole expression *e.g.* ${}^{2}\omega_2 = Z_{12}$. Then the later recursion steps will use the intermediate variables for their respective evaluations. This ensures that the computations specific to an antecedent frame are performed only once, and the result is reused in succeeding frame computations. The explicit forms are needed to preserve important quantities that will be needed to perform control and simulation of the model (*e.g.* the inertia matrix).

*CoM Terms*

In the Kane's formulation, velocities and accelerations of the CoM of each body $k$ ($\bar{v}_{Gk}$, $\bar{a}_{Gk}$) are needed. These will be evaluated using

$$\bar{v}_{Gk} = \bar{v}_k + \bar{\omega}_k \times \bar{r}_{Gk/O_k} \tag{3.23}$$

$$\bar{a}_{Gk} = \bar{a}_k + \bar{\alpha}_k \times \bar{r}_{Gk/O_k} + \bar{\omega}_k \times (\bar{\omega}_k \times \bar{r}_{Gk/O_k}) \tag{3.24}$$

The angular momenta terms are

$$\bar{H}_{Gk} = \mathbf{J}_{Gk}\bar{\omega}_k \tag{3.25}$$

$$\frac{d\bar{H}_{Gk}}{dt} = \mathbf{J}_{Gk}\bar{\alpha}_k + \bar{\omega}_k \times \mathbf{J}_{Gk}\bar{\omega}_k \tag{3.26}$$

where $\mathbf{J}_{Gk}$ is the inertia tensor of body $k$ around its CoM evaluated along the axes of frame fixed on $k$. If a given inertia tensor is around frame origin $O_k$ (denote by $\mathbf{J}_k$) then parallel-axis theorem will be used to find $\mathbf{J}_{Gk}$

$$\mathbf{J}_{Gk} = \mathbf{J}_k + m_k \mathbf{S}_k^{\times} \mathbf{S}_k^{\times}$$

where

$S_k$ Position vector $\bar{r}_{Gk/O_k}$ represented in frame $k$

$\mathbf{S}_k^{\times}$ Skew-symmetric matrix for cross product operation using $S_k$

*Partial Derivative* $(\bar{\omega}_k)_j = \frac{\partial \bar{\omega}_k}{\partial \dot{\mathbf{q}}_j}$

Tree structure allows a neat closed-form expression for the partial derivatives. For any body $k$ in the tree, its angular velocity can be expressed as

$$\bar{\omega}_k = \sum_{g \in \mathbb{G}_k} \dot{q}_g \bar{e}_g + \dot{q}_k \bar{e}_k \tag{3.27}$$

Partial differentiation with respect to each generalized speed $\dot{\mathbf{q}}_j$ is listed in (3.28)

| $\dot{\mathbf{q}}_j$ | $\frac{\partial \bar{\omega}_k}{\partial \dot{\mathbf{q}}_j}$ |
|---|---|
| $\dot{x}$ | $0$ |
| $\dot{q}_i \; i \in \mathbb{F}$ | $0 \quad \text{if } k \in \mathbb{G}_i$<br>$\bar{e}_i \quad \text{if } k \in \{i \cup \mathbb{F}_i\}$ |

(3.28)

The results in (3.28) follow from the fact that $\bar{\omega}_k$ in (3.27) does not depend on $\dot{x}$ or generalized speeds associated with succeeding frames $i \in \mathbb{F}_k$ ($\Longleftrightarrow k \in \mathbb{G}_i$). However, it depends on generalized speeds of antecedent frames and its own generalized speeds $\dot{q}_i \;\; i \in \{\mathbb{G}_k \cup k\}$ ($\Longleftrightarrow k \in \{i \cup \mathbb{F}_i\}$).

*Partial Derivative* $(\bar{v}_{Gk})_j = \frac{\partial \bar{v}_{Gk}}{\partial \dot{\mathbf{q}}_j}$

Linear velocity of body $k$ in the tree structure is given by

$$\bar{v}_{Gk} = \bar{v}_0 + \sum_{g \in \mathbb{G}_k} \left( \bar{\omega}_g \times \bar{r}_{O_{g^+}/O_g} \right) + \bar{\omega}_k \times \bar{r}_{Gk/O_k} \tag{3.29}$$

where $g^+ \in \{\mathbb{G}_k \cup k\}$ refers to the frame immediately succeeding $g$ in the chain. Partial differentiation with respect to each generalized speed $\dot{\mathbf{q}}_j$ is listed in (3.30)

| $\dot{\mathbf{q}}_j$ | $\frac{\partial \bar{v}_{Gk}}{\partial \dot{\mathbf{q}}_j}$ | |
|---|---|---|
| $\dot{x}$ | $\bar{i}_0$ | |
| $\dot{q}_i \; i \in \mathbb{F}$ | $0$ | if $k \in \mathbb{G}_i$ |
| | $\bar{e}_i \times \bar{r}_{Gk/O_i}$ | if $k \in \{i \cup \mathbb{F}_i\}$ |

(3.30)

The results in (3.30) follow from the fact that $\bar{v}_0 = \dot{x}\bar{i}_0$ is the only term in (3.29) that depends on $\dot{x}$. Moreover, owing to (3.28), partial derivatives wrt succeeding frame speeds $\dot{q}_i \;\; k \in \mathbb{G}_i$ vanish as none of the terms in (3.29) depend on them. With respect to antecedent/current frame speeds $\dot{q}_i \; k \in \{i \cup \mathbb{F}_i\}$, only those terms in the summation in (3.29) yield non-zero terms for which $g \in \{i \cup \mathbb{F}_i\}$, owing again to (3.28). Simplification performed in this case results in the expression listed in (3.30) as described below

$$\begin{aligned}
\frac{\partial \bar{v}_{Gk}}{\partial \dot{q}_i} &= \sum_{g \in \{i \cup \mathbb{F}_i\} \cap \mathbb{G}_k} \left(\bar{e}_i \times \bar{r}_{O_{g+}/O_g}\right) + \bar{e}_i \times \bar{r}_{Gk/O_k} \\
&= \bar{e}_i \times \left( \sum_{g \in \{i \cup \mathbb{F}_i\} \cap \mathbb{G}_k} \bar{r}_{O_{g+}/O_g} + \bar{r}_{Gk/O_k} \right) \\
&= \bar{e}_i \times \bar{r}_{Gk/O_i} \qquad k \in \{i \cup \mathbb{F}_i\}
\end{aligned} \tag{3.31}$$

*Tree-Structure Contribution to Kane's LHS*

The contribution of bodies in the tree structure to the LHS of the ODE corresponding to each generalized speed $\dot{\mathbf{q}}_j$ is listed in (3.32)

| $\dot{\mathbf{q}}_j$ | $\sum_{k\in\mathbb{F}_0} m_k \bar{a}_{Gk} \cdot (\bar{v}_{Gk})_j + \left(\frac{d\bar{H}_{Gk}}{dt}\right) \cdot (\bar{\omega}_k)_j$ |
|---|---|
| $\dot{x}$ | $\sum_{k\in\mathbb{F}_0} m_k \bar{a}_{Gk} \cdot \bar{i}_0$ |
| $\dot{q}_i \; i \in \mathbb{F}$ | $\sum_{k\in\{i\cup\mathbb{F}_i\cap\mathbb{F}_0\}} \Big[ \left(\bar{r}_{Gk/O_i} \times m_k \bar{a}_{Gk}\right) \cdot \bar{e}_i + \left(\mathbf{J}_{Gk}\bar{\alpha}_k + \bar{\omega}_k \times \mathbf{J}_{Gk}\bar{\omega}_k\right) \cdot \bar{e}_i \Big]$ |

(3.32)

The results in (3.32) follow from (3.26), (3.28) and (3.30), as well as from the following manipulation using a triple-product identity

$$m_k \bar{a}_{Gk} \cdot \left(\bar{e}_i \times \bar{r}_{Gk/O_i}\right) = \left(\bar{r}_{Gk/O_i} \times m_k \bar{a}_{Gk}\right) \cdot \bar{e}_i$$

Physical interpretation of the terms in (3.32) is simple. Tree-structure contributions to LHS of the $\dot{x}$ equation is simply the sum of inertial forces resulting from body accelerations projected onto the direction of $\dot{x}$ *i.e.*, $\bar{i}_0$. Contributions to equations corresponding to generalized speeds $\dot{q}_i \; i \in \mathbb{F}$, are simply evaluated by combining the inertial torques that result from linear and angular motions of all bodies $k \in \{i \cup \mathbb{F}_i \cap \mathbb{F}_0\}$, meaning the body which $\dot{q}_i$ corresponds to and all bodies that succeed $i$ in the tree structure. Set intersection with $\mathbb{F}_0$ simply means that frame $0$ is excluded from the summation, which is because $0$ is not a label to a body in the system. This has effect in only the $\dot{q}_0 = \dot{\psi}$ equation where summation will start from $k = 1$, as opposed to all other $\dot{q}_i$ equations where summation will not exclude inertial terms for $k = i$..

## 3.5 Kane's Right-Hand Side

To evaluate the right hand side of the Kane's forumulation (3.8), external forces and torques on the system are

$\bar{\tau}_L, \bar{\tau}_R$ Torques on left and right wheels by motors fixed on the base link

$\bar{\tau}_j = \tau_j \bar{e}_j \;\; j \in \mathbb{F}_0$ Torques on bodies in the tree structure. These are applied by their respective joint motors, except $\bar{\tau}_1$ which is the torque on base link in reaction to the wheel torques $\bar{\tau}_L$ and $\bar{\tau}_R$ *i.e.*, $\bar{\tau}_1 = -\bar{\tau}_R - \bar{\tau}_L$

$-\bar{\tau}_j = -\tau_j \bar{e}_j \;\; j \in \mathbb{F}_1$ Reaction torques experienced by antecedent links $j^-$ in the tree structure. $-\bar{\tau}_1$ is excluded because it signifies wheel torques $\bar{\tau}_L, \bar{\tau}_R$, already covered

$\bar{F}_{Gk} = m_k \bar{g}$ is the weight of each joint $k$

$\bar{F}_{El}, \bar{\tau}_{El}$ Force and torque applied by the environment on the left hand end-effector at point $E_l$

$\bar{F}_{Er}, \bar{\tau}_{Er}$ Force and torque applied by the environment on the right hand end-effector at point $E_r$

### 3.5.1 Wheel Torques

The contribution on the right-hand side of Kane's equations of wheel motor torques $\bar{\tau}_L = \tau_L \bar{j}_0$ and $\bar{\tau}_R = \tau_R \bar{j}_0$ and their reaction $\bar{\tau}_1 = -\bar{\tau}_R - \bar{\tau}_L$ is listed in (3.33)

| $\dot{\mathbf{q}}_\mathbf{j}$ | $\bar{\tau}_L \cdot \frac{\partial \bar{\omega}_L}{\partial \dot{\mathbf{q}}_j} + \bar{\tau}_R \cdot \frac{\partial \bar{\omega}_R}{\partial \dot{\mathbf{q}}_j} + \bar{\tau}_1 \cdot \frac{\partial \bar{\omega}_1}{\partial \dot{\mathbf{q}}_j}$ |
|---|---|
| $\dot{x}$ | $\frac{1}{R}(\tau_L + \tau_R)$ |
| $\dot{\psi}$ | $\frac{L}{2R}(\tau_R - \tau_L)$ |
| $\dot{q}_1$ | $-(\tau_L + \tau_R)$ |
| $\dot{q}_i \; i \in \mathbb{F}_1$ | $0$ |

(3.33)

The results in (3.33) follow from partial derivatives listed in (3.15) and from $\bar{\omega}_1 = \dot{\psi}\bar{k}_0 + \dot{q}_1\bar{j}_0$.

### 3.5.2 Joint Torques

Contributions to to Kane's RHS of joint motor torques and their reactions is listed in (3.34)

| $\dot{\mathbf{q}}_\mathbf{j}$ | $\sum_{k\in\mathbb{F}_1}\left(\tau_k\bar{e}_k\cdot\frac{\partial\bar{\omega}_k}{\partial\dot{\mathbf{q}}_j}-\tau_k\bar{e}_k\cdot\frac{\partial\bar{\omega}_{k-}}{\partial\dot{\mathbf{q}}_j}\right)$ |
|---|---|
| $\dot{x}$ | $0$ |
| $\dot{\psi}$ | $0$ |
| $\dot{q}_1$ | $0$ |
| $\dot{q}_i \in \mathbb{F}_1$ | $\tau_i$ |

(3.34)

The results in (3.34) follow from (3.28). Based on (3.28), we readily deduce that, for a given $\dot{q}_i$ $i \in \mathbb{F}$, the term in summation will be 0 for $k \in \mathbb{G}_i$. As for $k \in \mathbb{F}_i$, the term becomes $\tau_k\bar{e}_k\cdot\bar{e}_i - \tau_k\bar{e}_k\cdot\bar{e}_i = 0$. Only for the case of $k = i$, the term is $\tau_i\bar{e}_i\cdot\bar{e}_i - 0 = \tau_i$.

### 3.5.3 Gravitational Forces

Weight of wheels $-m_w g\bar{k}_0$ is perpendicular to wheel velocities $\bar{v}_{GL}$ and $\bar{v}_{GR}$ (see (3.10) and (3.12)) therefore does not feature in the final dynamics ODEs. The contribution of the weight of bodies $k \in \mathbb{F}_0$ in the tree structure are listed in (3.35)

| $\dot{\mathbf{q}}_j$ | $\sum_{k\in\mathbb{F}_0} m_k\bar{g}\cdot\frac{\partial\bar{v}_{Gk}}{\partial\dot{\mathbf{q}}_j}$ |
|---|---|
| $\dot{x}$ | $0$ |
| $\dot{\psi}$ | $0$ |
| $\dot{q}_i$ $i \in \mathbb{F}_0$ | $\sum_{k\in\{i\cup\mathbb{F}_i\}}\left(\bar{r}_{Gk/O_i}\times m_k\bar{g}\right)\cdot\bar{e}_i$ |

(3.35)

The results in (3.35) follow from (3.30). For $\dot{x}$, partial derivative is $\bar{i}_0$, which is normal to the weights causing the dot products to vanish. For $\dot{q}_i$ $i \in \mathbb{F}$, we refer the reader to the explanation for (3.32). For $\dot{\psi}$, the resulting expression vanishes due to cross product of

weight $\bar{g} = -g\bar{k}_0$ with $\bar{k}_0$ as shown below.

$$\begin{aligned}\left(\bar{r}_{Gk/O_i} \times m_k\bar{g}\right) \cdot \bar{e}_0 &= \left(\bar{r}_{Gk/O_i} \times m_k\bar{g}\right) \cdot \bar{k}_0 \\ &= \left(m_k\bar{g} \times \bar{k}_0\right) \cdot \bar{r}_{Gk/O_i} \\ &= 0\end{aligned}$$

Physical interpretation of (3.35) is that gravitational forces do not directly affect forward motion $\dot{x}$ and spin $\dot{\psi}$ dynamics. As for joints $i \in \mathbb{F}_0$ in the tree structure, RHS contributions are calculated by projecting the sum of weight torques of current/succeeding links $k \in \{i \cup \mathbb{F}_i\}$ on the joint rotation axis.

### 3.5.4 End-effector Forces and Torques

Contributions of contact wrenches $\bar{F}_{Ea}, \bar{\tau}_{Ea}$ $a \in \{l, r\}$ on left and right arm end-effectors to RHS of Kane's equations is listed in (3.36)

| $\dot{\mathbf{q}}_j$ | $\bar{F}_{Ea} \cdot \frac{\partial \bar{v}_{Ea}}{\partial \dot{\mathbf{q}}_j} + \bar{\tau}_{Ea} \cdot \frac{\partial \bar{\omega}_{10a}}{\partial \dot{\mathbf{q}}_j}$ $a \in \{l, r\}$ |
|---|---|
| $\dot{x}$ | $\bar{F}_{Ea} \cdot \bar{i}_0$ |
| $\dot{q}_i$ $i \in \mathbb{F}$ | $0 \quad i \notin \{\mathbb{G}_{10a} \cup 10a\}$ |
| | $\left(\bar{r}_{Ea/O_i} \times \bar{F}_{Ea} + \bar{\tau}_{Ea}\right) \cdot \bar{e}_i \quad i \in \{\mathbb{G}_{10a} \cup 10a\}$ |

(3.36)

The results in (3.36) follow from (3.28) applied to the following expression for end-effector velocity, the details of which can be infered from the explanation for (3.30).

$$\bar{v}_{Ea} = \bar{v}_0 + \sum_{k \in \mathbb{G}_{10a}} \left(\bar{\omega}_k \times \bar{r}_{O_{k+}/O_k}\right) + \bar{\omega}_{10a} \times \bar{r}_{Ea/O_{10a}} \; a \in \{l, r\}$$

Physically the interpretation of (3.36) is that for $\dot{x}$, the exernal force is projected on $\bar{i}_0$, and for joint speeds $\dot{q}_i$ $i \in \mathbb{F}$ torque from external force about the joint origin is projected on

$$
\begin{array}{lcccccc}
 & \text{Body Link Motions} & & \text{Wheel Motion} & & \text{Actuator Torques} & & \text{End-Effector Interactions} \\
\dot{x}: & \sum_{k\in\mathbb{F}_0}(m_k\bar{a}_{Gk})\cdot\bar{i}_0 & + & \left(2m_w+\frac{2\mathbf{Y}\mathbf{Y}_w}{R^2}\right)\ddot{x} & = & \frac{1}{R}(\tau_L+\tau_R) & + & \sum_{a\in\mathbb{A}}\bar{F}_{Ea}\cdot\bar{i}_0 \\
\dot{\psi}: & \sum_{k\in\mathbb{F}_0}\left(\bar{r}_{Gk/O_0}\times m_k\bar{a}_{Gk}+\mathbf{J}_{Gk}\bar{\alpha}_k+\bar{\omega}_k\times\mathbf{J}_{Gk}\bar{\omega}_k\right)\cdot\bar{k}_0 & + & \left(\frac{m_wL^2}{2}+\frac{\mathbf{Y}\mathbf{Y}_wL^2}{2R^2}+2\mathbf{Z}\mathbf{Z}_w\right)\ddot{\psi} & = & \frac{L}{2R}(\tau_R-\tau_L) & + & \sum_{a\in\mathbb{A}}\left(\bar{r}_{Ea/O_0}\times\bar{F}_{Ea}+\bar{\tau}_{Ea}\right)\cdot\bar{k}_0 \\
\dot{q}_1: & \sum_{k\in\mathbb{F}_0}\left(\bar{r}_{Gk/O_1}\times m_k\bar{a}_{Gk}+\mathbf{J}_{Gk}\bar{\alpha}_k+\bar{\omega}_k\times\mathbf{J}_{Gk}\bar{\omega}_k\right. & - & \left.\bar{r}_{Gk/O_1}\times m_k\bar{g}\right)\cdot\bar{j}_0 & = & -(\tau_L+\tau_R) & + & \sum_{a\in\mathbb{A}}\left(\bar{r}_{Ea/O_1}\times\bar{F}_{Ea}+\bar{\tau}_{Ea}\right)\cdot\bar{j}_0 \\
\dot{q}_j: & \sum_{k\in\{j\cup\mathbb{F}_j\}}\left(\bar{r}_{Gk/O_j}\times m_k\bar{a}_{Gk}+\mathbf{J}_{Gk}\bar{\alpha}_k+\bar{\omega}_k\times\mathbf{J}_{Gk}\bar{\omega}_k\right. & - & \left.\bar{r}_{Gk/O_j}\times m_k\bar{g}\right)\cdot\bar{e}_j & = & \tau_j & + & \sum_{a\in\mathbb{A}}\mathbb{I}_j(a)\left(\bar{r}_{Ea/O_j}\times\bar{F}_{Ea}+\bar{\tau}_{Ea}\right)\cdot\bar{e}_j \\
j\in\mathbb{F}_1 & & & \text{Body Link Weights} & & & & 
\end{array}
\tag{3.37}
$$

Table 3.1: Equations of motion of the full dynamic model

the joint axis to evaluate the effects on the respective speed dynamics. This is applicable to the link which is part of the chain in the antecedent link of the end-effector $\{\mathbb{G}_{10a}\cup 10a\}$. Define an indicator function $\mathbb{I}_j(a)$ as 1 if $j\in\{\mathbb{G}_{10a}\cup 10a\}$ and 0 otherwise. Then the result for the last row for both arms $\mathbb{A}=\{l,r\}$ can be written in compact form as $\sum_{a\in\mathbb{A}}\mathbb{I}_j(a)\left(\bar{r}_{Ea/O_j}\times\bar{F}_{Ea}+\bar{\tau}_{Ea}\right)\cdot\bar{e}_j$.

The resulting full form that the dynamic model takes is listed in (3.37).

## 3.6 Zero Dynamics

Evaluating the dynamics equations (3.37) and assuming zero end-effector wrench, the resulting dynamics can be written as:

$$
A(\mathbf{q})\ddot{\mathbf{q}}+C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}+Q(\mathbf{q})=\begin{pmatrix}-\tau_1/R\\ \\ \Gamma\end{pmatrix}+\Gamma_{fric} \tag{3.38}
$$

$A(\mathbf{q})\in\mathbb{R}^{19\times 19}$ is the inertia matrix. $C(\mathbf{q},\dot{\mathbf{q}})\in\mathbb{R}^{19\times 19}$ represents the Coriolis effects, $Q(\mathbf{q})\in\mathbb{R}^{19\times 1}$ represents gravitational effects, $\Gamma\in\mathbb{R}^{18\times 1}$ is the vector of torques $\tau_j$ $j\in\mathbb{F}$

in the same order as $q_j$ in $\mathbf{q}$. Here we defined

$$\tau_0 = \frac{L}{2R}(\tau_R - \tau_L)$$
$$\tau_1 = -(\tau_L + \tau_R)$$

$\Gamma_{fric} \in \mathbb{R}^{19\times 1}$ represents the frictional effects. Note that both $\dot{x}$ and $\dot{q}_1$ are actuated by the same torque $\tau_1$. Eliminating this coupling torque from the two equations results in zero dynamics of the system. Before we proceed to analyse the zero dynamics, we need to find expressions for elements of inertia matrix $A$ in the rows corresponding to $\dot{x}$ and $\dot{q}_1$ equations, and also corresponding elements of $Q$ vector.

### 3.6.1 Elements of Inertia Matrix $A$

Elements of $A$ in (3.38) are coefficients of $\ddot{\mathbf{q}}_j$ in (3.37). Since the terms in (3.37) are linear in $\ddot{\mathbf{q}}$, taking partial derivatives of LHS terms in (3.37) wrt $\ddot{\mathbf{q}}_j$ will give us the coefficients of $\ddot{\mathbf{q}}_j$.

*Partial Derivative* $\frac{\partial \bar{\alpha}_k}{\partial \ddot{\mathbf{q}}}$

Using (3.18), it can be deduced that $\bar{\alpha}_k$ $k \in \mathbb{F}_0$ can be expressed as

$$\bar{\alpha}_k = \sum_{g\in\mathbb{G}_k} \ddot{q}_g \bar{e}_g + \ddot{q}_k \bar{e}_k + \bar{\eta}(\dot{\mathbf{q}}) \tag{3.39}$$

$\bar{\eta}$ denotes a (vector) function of $\dot{q}_j$. Since (3.39) takes the same form as (3.27), the reasoning for (3.28) quickly gives us the results listed in (3.40)

| $\ddot{\mathbf{q}}_j$ | $\frac{\partial \bar{\alpha}_k}{\partial \ddot{\mathbf{q}}_j}$ |
|---|---|
| $\ddot{x}$ | $0$ |
| $\ddot{q}_i \; i \in \mathbb{F}$ | $0 \quad \text{if } k \in \mathbb{G}_i$<br>$\bar{e}_i \quad \text{if } k \in \{i \cup \mathbb{F}_i\}$ |

(3.40)

*Partial Derivative* $\frac{\partial \bar{a}_{Gk}}{\partial \ddot{\mathbf{q}}}$

Using (3.18), it can be deduced that $\bar{a}_{Gk} \; k \in \mathbb{F}_0$ can be expressed as

$$\bar{a}_{Gk} = \bar{a}_0 + \sum_{g \in \mathbb{G}_k} \left( \bar{\alpha}_g \times \bar{r}_{O_{g+}/O_g} \right) + \bar{\alpha}_k \times \bar{r}_{Gk/O_k} + \bar{\eta}(\dot{\mathbf{q}}) \tag{3.41}$$

Similarity with (3.29) allows us to invoke the reasoning for (3.30) to quickly deduce results in (3.42)

| $\ddot{\mathbf{q}}_j$ | $\frac{\partial \bar{a}_{Gk}}{\partial \ddot{\mathbf{q}}_j}$ |
|---|---|
| $\ddot{x}$ | $\bar{i}_0$ |
| $\ddot{q}_i \; i \in \mathbb{F}$ | $0 \quad \text{if } k \in \mathbb{G}_i$<br>$\bar{e}_i \times \bar{r}_{Gk/O_i} \quad \text{if } k \in \{i \cup \mathbb{F}_i\}$ |

(3.42)

*Partial Derivatives of* $\dot{x}$ *Equation in* (3.37)

For $\dot{x}$ equation in (3.37), the results are listed in (3.43). They inform us about the elements of the row of inertia matrix corresponding to $\ddot{x}$ equation (first row of $A$). Following new notations have been introduced in (3.43): Body mass $M = \sum_{k \in \mathbb{F}_0} m_k$ and body CoM frame

0 coordinates $[X_{com}, Y_{com}, Z_{com}]^\top = \left( \sum_{k\in\mathbb{F}_0} m_k \bar{r}_{Gk/O_1} / \sum_{k\in\mathbb{F}_0} m_k \right) \cdot [\bar{i}_0, \bar{j}_0, \bar{k}_0]^\top$.

| Element of $A$ | $\ddot{\mathbf{q}}_j$ | $\frac{\partial}{\partial \ddot{\mathbf{q}}_j} \left( \sum_{k\in\mathbb{F}_0} (m_k \bar{a}_{Gk}) \cdot \bar{i}_0 + \left(2m_w + \frac{2\mathbf{YY}_w}{R^2}\right) \ddot{x} \right)$ |
|---|---|---|
| $a_{xx}$ | $\ddot{x}$ | $2m_w + \frac{2\mathbf{YY}_w}{R^2} + M$ |
| $a_{x(q0)}$ | $\ddot{\psi}$ | $-MY_{com}$ |
| $a_{x(q1)}$ | $\ddot{q}_1$ | $MZ_{com}$ |
| $a_{x(qi)}$ | $\ddot{q}_i \; i \in \mathbb{F}_1$ | $(\bar{i}_0 \times \bar{e}_i) \cdot \sum_{k\in\{i\cup\mathbb{F}_i\}} m_k \bar{r}_{Gk/O_i}$ General Case<br>$M_i Z_{com(i)}$ Planar Case |

(3.43)

Results in (3.43) follow easily from (3.42), and from the triple product rule applied to last three results. The expression in the general case of the last row applies as well to $\ddot{\psi}$ and $\ddot{q}_1$, but the explicit knowledge of $\bar{e}_i$ in both cases allows us to give a more simplified and meaningful expression. Physically, for $\ddot{x}$, the term represents that all masses in the system have been added along with inertial effects of the wheel. For $\ddot{\psi}$ and $\ddot{q}_1$, the terms represent respectively the $-y$ and $z$ components of body's first moment around wheel axis. If we had a planar robot then $\bar{e}_i = \bar{j}_0 \; \forall i \in \mathbb{F}_1$, allowing us to simplify the general expression of the last row to the one for the planar case where $M_i = \sum_{k\in\{i\cup\mathbb{F}_i\}} m_k$ is the mass of the articulated structure on joint $i$, and $Z_{com(i)} = \left( \sum_{k\in\{i\cup\mathbb{F}_i\}} m_k \bar{r}_{Gk/O_i} / \sum_{k \; in\{i\cup\mathbb{F}_i\}} m_k \right) \cdot \bar{k}_0$ is $z$ component of its CoM in frame 0. The final term $M_i Z_{com(i)}$ represents the first moment of the articulated structure on joint $i$ around wheel axis.

*Partial Derivatives of $\dot{q}_1$ Equation in* (3.37)

For $\dot{q}_1$ equation in (3.37), the results are listed in (3.45). They inform us about the elements of the row of inertia matrix corresponding to $\ddot{q}_1$ equation (third row of $A$). We denote with $I$ the moment of inertia of the full body around wheel axis, found by applying parallel axis theorem

$$I = \sum_{k \in \mathbb{F}_0} \left[ (\mathbf{J}_{Gk} \bar{j}_0) \cdot \bar{j}_0 + m_k (r_{Gk/O_1} \cdot \bar{i}_0)^2 + m_k (r_{Gk/O_1} \cdot \bar{k}_0)^2 \right] \tag{3.44}$$

| Element of $A$ | $\ddot{\mathbf{q}}_j$ | $\frac{\partial}{\partial \ddot{\mathbf{q}}_j} \sum_{k \in \mathbb{F}_0} \left( \bar{r}_{Gk/O_1} \times m_k \bar{a}_{Gk} + \mathbf{J}_{Gk} \bar{\alpha}_k \right) \cdot \bar{j}_0$ |
|---|---|---|
| $a_{(q1)x}$ | $\ddot{x}$ | $MZ_{com}$ |
| $a_{(q1)(q0)}$ | $\ddot{\psi}$ | $\sum_{k \in \mathbb{F}_0} \left( \bar{r}_{Gk/O_1} \times m_k (\bar{k}_0 \times \bar{r}_{Gk/O_1}) + \mathbf{J}_{Gk} \bar{k}_0 \right) \cdot \bar{j}_0$ |
| $a_{(q1)(q1)}$ | $\ddot{q}_1$ | $I$ Body Inertia |
| $a_{(q1)(qi)}$ | $\ddot{q}_i \; i \in \mathbb{F}_1$ | $\sum_{k \in \{i \cup \mathbb{F}_i\}} \left( \bar{r}_{Gk/O_1} \times m_k (\bar{e}_i \times \bar{r}_{Gk/O_i}) + \mathbf{J}_{Gk} \bar{e}_i \right) \cdot \bar{j}_0$ $= \beta_i$ (3.46) Planar Case |

(3.45)

The results in (3.45) follow easily from (3.40) and (3.42). Unsurprisingly, the term for $a_{(q1)x}$ is the same as that for $a_{x(q1)}$, as $A$ is supposed to be symmetric. Term for $a_{(q1)(q1)}$ is the total body inertia around wheel axis (3.44). Other terms $a_{(q1)(qi)}$ $i \in \mathbb{F}$ represent coupled inertial effects accounting for the inertial torque about wheel axis $\bar{j}_0$ as the articulated structure on $i$th joint accelerates around $\bar{e}_i$. If we had a planar robot then $\bar{e}_i = \bar{j}_0 \; \forall i \in \mathbb{F}_1$, allowing us

to simplify the general expression of the last row to

$$\begin{aligned}\beta_i = \sum_{k\in\{i\cup\mathbb{F}_i\}} \Big[ & (\mathbf{J}_{Gk}\bar{j}_0)\cdot\bar{j}_0 \\ & + m_k(\bar{r}_{Gk/O_1}\cdot\bar{i}_0)(\bar{r}_{Gk/O_i}\cdot\bar{i}_0) + m_k(\bar{r}_{Gk/O_1}\cdot\bar{k}_0)(\bar{r}_{Gk/O_i}\cdot\bar{k}_0)\Big]\end{aligned} \tag{3.46}$$

### 3.6.2 Components of Gravitational Vector $Q$

To derive the zero dynamics, we are interested in $Q_x$ and $Q_{q1}$, the elements of $Q$ vector in (3.38). Since gravitational forces have no contribution in $\dot{x}$ equation in (3.37),

$$Q_x = 0$$

As for $Q_{q1}$, we evaluate the gravity terms in $\dot{q}_1$ equation in (3.37) by making use of $\bar{g} = -g\bar{k}_0$ and the triple product rule, and defining $M = \sum_{k\in\mathbb{F}_0} m_k$

$$\begin{aligned} Q_{q1} &= -\sum_{k\in\mathbb{F}_0}\left(\bar{r}_{Gk/O_1}\times m_k\bar{g}\right)\cdot\bar{j}_0 \\ &= -\bar{i}_0\cdot\sum_{k\in\mathbb{F}_0} m_k g\bar{r}_{Gk/O_1} \\ &= -Mg\left(\frac{\sum_{k\in\mathbb{F}_0} m_k\bar{r}_{Gk/O_1}}{\sum_{k\in\mathbb{F}_0} m_k}\right)\cdot\bar{i}_0 \\ &= -MgX_{com} \end{aligned} \tag{3.47}$$

where we used $X_{com}$ to define the $x$ component of body's CoM in frame $0$. Physically, (3.47) is representing the weight torque of the full body around wheel axis.

### 3.6.3 Zero Dynamics

We find the zero dynamics of the robot by eliminating $\tau_1$ from the $\dot{x}$ and $\dot{q}_1$ equations in the dynamic model. The two equations of the dynamic model are

$$\begin{pmatrix} a_{xx} & a_{xq}^\top \\ a_{x(q1)} & a_{(q1)q}^\top \end{pmatrix} \begin{pmatrix} \ddot{x} \\ \ddot{q} \end{pmatrix} + \begin{pmatrix} C'_x \\ C'_{q_1} \end{pmatrix} + \begin{pmatrix} Q_x \\ Q_{q_1} \end{pmatrix} = \begin{pmatrix} -\frac{\tau_1}{R} \\ \tau_1 \end{pmatrix} \tag{3.48}$$

where $\begin{bmatrix} a_{xx} & a_{xq}^\top \end{bmatrix}$ and $\begin{bmatrix} a_{x(q1)} & a_{(q1)q}^\top \end{bmatrix}$ are the first two rows of inertia matrix $A$. We also introduced $q$ (non-bold-face and without subscripts) as the vector of joint coordinates *i.e.*, $\mathbf{q} = \begin{bmatrix} x & q^\top \end{bmatrix}^\top$. $\begin{bmatrix} C'_x & C'_{q_1} \end{bmatrix}^\top$ and $\begin{bmatrix} Q_x & Q_{q_1} \end{bmatrix}^\top$ are the first two elements of $C'$ and $Q$ respectively. Here we have lumped together Coriolis and frictional effects into $C'$ matrix as

$$C' = C \begin{pmatrix} \dot{x} \\ \dot{q} \end{pmatrix} - \Gamma_{fric}$$

Eliminating $\tau_1$ from the two equations results in

$$\begin{aligned} &(Ra_{xx} + a_{x(q1)})\ddot{x} + \left(Ra_{xq}^\top + a_{(q1)q}^\top\right)\ddot{q} \\ &\quad + \left(RC'_x + C'_{q_1}\right) + (RQ_x + Q_{q_1}) = 0 \end{aligned}$$

$$\Rightarrow \left[R\left(2m_w + \frac{2\mathbb{Y}\mathbb{Y}_w}{R^2} + M\right) + MZ_{com}\right]\ddot{x} \tag{3.49}$$

$$+ \left[R\begin{pmatrix} MZ_{com} \\ a_{x(q2)} \\ \vdots \\ a_{x(q10r)} \end{pmatrix}^\top + \begin{pmatrix} I \\ a_{(q1)(q2)} \\ \vdots \\ a_{(q1)(q10r)} \end{pmatrix}^\top\right]\ddot{q}$$

$$+ \left(RC'_x + C'_{q_1}\right) - MgX_{com} = 0 \tag{3.50}$$

For the planar case, this can be expressed as

$$\Rightarrow \left[R\left(2m_w + \frac{2\mathbb{Y}\mathbb{Y}_w}{R^2} + M\right) + MZ_{com}\right]\ddot{x} \tag{3.51}$$
$$+ \left[R\begin{pmatrix} MZ_{com} \\ M_2 Z_{com(2)} \\ \vdots \\ M_{10r} Z_{com(10r)} \end{pmatrix}^{\top} + \begin{pmatrix} I \\ \beta_2 \\ \vdots \\ \beta_{10r} \end{pmatrix}^{\top}\right]\ddot{q}$$
$$+ \left(RC'_x + C'_{q_1}\right) - MgX_{com} = 0 \tag{3.52}$$

### 3.7 Simplified Model

For high-level controller, we propose to represent the full robot as a Wheeled Inverted Pendulum Model (WIPM)—a one-link robot of equivalent inertial properties dynamically balancing itself on the wheel. Results from Kane's analysis can help us quickly derive its model. We use $\theta$ to represent the pitch of the link. For the full robot, $\theta$ is the angular position of the body center of mass. Using $\dot{x}$ and $\dot{q}_1$ equations in (3.37), the model of WIPM is:

$$\begin{aligned} \left(m_w + \frac{I_w}{R^2}\right)\ddot{x} + M\bar{a}_{G1}\bar{i}_0 &= \frac{-\tau_1}{R} \\ \left(\bar{r}_{G1} \times M\bar{a}_{G1} + \mathbf{J}_{G1}\bar{\alpha}_1 + \bar{\omega}_1 \times \mathbf{J}_{G1}\bar{\omega}_1 - \bar{r}_{G1} \times M\bar{g}\right) \cdot \bar{j}_0 &= \tau_1 \end{aligned} \tag{3.53}$$

Using

$$\begin{aligned} \bar{\omega}_1 &= \dot{\theta}\bar{j}_0 \\ \bar{\alpha}_1 &= \ddot{\theta}\bar{j}_0 \\ \bar{r}_{G1/O_0} &= X_{com}\bar{i}_0 + Z_{com}\bar{k}_0 \\ \mathbf{J}_{G1} &= I - M(X_{com}^2 + Z_{com}^2) \\ \bar{a}_{G1} &= \bar{a}_0 + (\bar{\alpha}_1 \times \bar{r}_{G1/O_0}) + \bar{\omega}_1 \times (\bar{\omega}_1 \times \bar{r}_{G1/O_0}) \\ &= \left(\ddot{x} + Z_{com}\ddot{\theta} - X_{com}\dot{\theta}^2\right)\bar{i}_0 + \left(-X_{com}\ddot{\theta} - Z_{com}\dot{\theta}^2\right)\bar{k}_0 \end{aligned} \tag{3.54}$$

(3.53) becomes

$$\begin{aligned} R\Big(m_w + \frac{I_w}{R^2} + M\Big)\ddot{x} + RMZ_{com}\ddot{\theta} + RMX_{com}\dot{\theta}^2 &= -\tau_1 \\ MZ_{com}\ddot{x} + I\ddot{\theta} - MX_{com}g &= \tau_1 \end{aligned} \tag{3.55}$$

Eliminating $\tau_1$ from (3.55) gives us the zero dynamics of the simplified model

$$\begin{aligned} &\Big[R\Big(m + \frac{I_w}{R^2} + M\Big) + MZ_{com}\Big]\ddot{x} \\ &\quad + (MZ_{com} + I)\,\ddot{\theta} \\ &\quad + RMX_{com}\dot{\theta}^2 - MgX_{com} = 0 \end{aligned} \tag{3.56}$$

## 3.8 Conclusion

In this material, we evaluated the dynamic model of a typical WIP humanoid. We used Kane's formulation for the task because of the use of quasi-velocity in our generalized velocity set that prevents us from using Lagrange formulation. The dynamic model can be used in designing of control systems and simulation of the robot. An accurate simulation will require a few steps that are not performed in this report: The accurate estimation of masses, center of masses, and inertial parameters of each link, accurate modeling of the current-torque relationship of the joint motors (prior experience tells us that the relationship is highly non-linear) and the role of friction in the dynamic model. These are challenging tasks yet necesssary if near-accurate numerical estimation of the dynamic model is to be obtained.

# CHAPTER 4
# FULL-BODY CONTROL

In this chapter, we propose a method to isolate the manipulator/body dynamics from wheel dynamics such that existing techniques for full-body control of humanoids are applicable. We consider the case in which the base link is not passive, in that, the reaction of the torque driving the wheels is acting on the base link. This special feature of WIP dynamics challenges the straightforward application of existing whole-body control techniques, compared to fully actuated, constrained, or under-actuated systems with no DOFs sharing the actuators. A side-effect of isolating manipulator dynamics is the resulting asymmetry of the inertia matrix restricting full use of operational space tools. This is addressed by presenting a coordinate transformation that transforms the inertia matrix into a symmetric one. We present a detailed adaptation of [1] to control the transformed system. The adaptation of [2] is also briefly discussed. In the next chapter, the wheel dynamics is derived as a function of manipulator joint accelerations so that horizontal motion of the robot can be planned as a high-level task to be executed by the full-body controller, developed in this chapter. Simulation results are presented on a five degree of freedom planar robot having a 4-DOF serial arm mounted on wheels.

## 4.1 Dynamic Model of the Planar Case

As discussed earlier, the spin motion of the robot is not the main source of complexity in WIP dynamics so we will limit our discussion to the planar case (Figure 4.1). Extending this theory for the full 3D model will be part of our future work. Ignoring thus the spin motion $\dot{\psi}$ and evaluating the dynamics equations 3.6-3.9, the resulting dynamics can be

written as:

$$A\begin{pmatrix}\ddot{x}\\ \ddot{q}\end{pmatrix} + C\begin{pmatrix}\dot{x}\\ \dot{q}\end{pmatrix} + Q = \begin{pmatrix}{}^{-\tau_1}/_{R}\\ \Gamma\end{pmatrix} + \Gamma_{fric} \tag{4.1}$$

$A \in \mathbb{R}^{(n+1)\times(n+1)}$ is the inertia matrix. $C \in \mathbb{R}^{(n+1)\times(n+1)}$ represents the Coriolis effects, $Q \in \mathbb{R}^{(n+1)\times 1}$ represents gravitational effects, $\Gamma = \begin{pmatrix}\tau_1 & ... & \tau_n\end{pmatrix}^T$ is the vector of torques acting on each joint and $\Gamma_{fric} \in \mathbb{R}^{(n+1)\times 1}$ represents the frictional effects. Note that both $\dot{x}$ and $\dot{q}_1$ are actuated by the same torque $\tau_1$. This is the characteristic that makes the control of this system interesting.

## 4.2 Isolating Manipulator Dynamics

The key idea is to isolate the $\dot{q}$ dynamics by eliminating $\dot{x}$. This is achieved by first left-multiplying eq. (3.38) with $A^{-1}$.

$$\begin{pmatrix}\ddot{x}\\ \ddot{q}\end{pmatrix} = A^{-1}\Big(-C\begin{pmatrix}\dot{x}\\ \dot{q}\end{pmatrix} - Q + \Gamma_{fric}\Big) + A^{-1}\begin{pmatrix}{}^{-\tau_1}/_{R}\\ \Gamma\end{pmatrix} \tag{4.2}$$

It is important to now define the blocks in the inverted matrix that affect the two subsystems. So we define:

$$A = \begin{bmatrix}a_{xx} & a_{xq}^{\mathsf{T}}\\ a_{xq} & A_{qq}\end{bmatrix}, A^{-1} = \begin{bmatrix}\bar{a}_x\\ \bar{A}q\end{bmatrix} = \begin{bmatrix}\bar{a}_{xx} & \bar{a}_{xq}^{\mathsf{T}}\\ \bar{a}_{xq} & \bar{A}_{qq}\end{bmatrix}$$

where [66]

$$\bar{a}_{xx} = \frac{1}{a_{xx} - a_{xq}^{\mathsf{T}} A_{qq}^{-1} a_{xq}}, \tag{4.3}$$

$$\bar{a}_{xq} = -\bar{a}_{xx} A_{qq}^{-1} a_{xq} \tag{4.4}$$

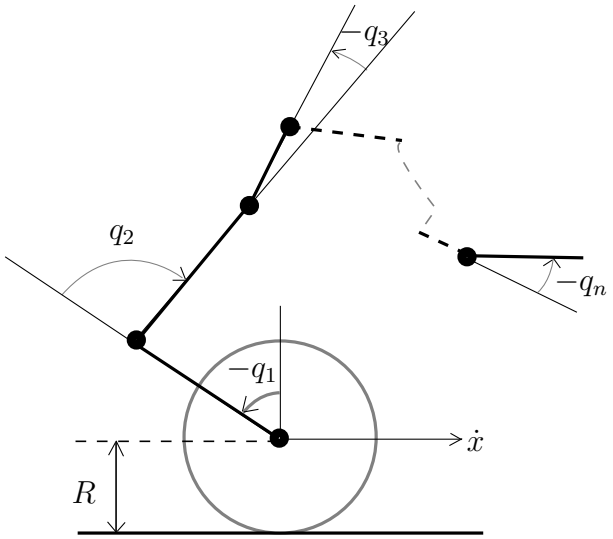



Figure 4.1: A Wheeled Inverted Pendulum Robot with $n$ joints

$$\bar{A}_{qq} = \left(A_{qq} - \frac{1}{a_{xx}} a_{xq} a_{xq}^{\mathsf{T}}\right)^{-1} \tag{4.5}$$

$$= \left(A_{qq}^{*}\right)^{-1} \tag{4.6}$$

A useful property of these matrices that we will use later is

$$A_{qq}^{*} \bar{a}_{xq} = -\frac{a_{xq}}{a_{xx}} \tag{4.7}$$

We can now rewrite eq. (4.2) as:

$$\begin{pmatrix} \ddot{x} \\ \\ \ddot{q} \end{pmatrix} = \begin{pmatrix} \bar{a}_x \\ \\ \bar{A}_q \end{pmatrix} (C' - Q) + \begin{pmatrix} \bar{a}_{xx} & \bar{a}_{xq}^{\mathsf{T}} \\ \\ \bar{a}_{xq} & \bar{A}_{qq} \end{pmatrix} \begin{pmatrix} -\tau_1/R \\ \\ \Gamma \end{pmatrix} \tag{4.8}$$

where

$$C' = -C \begin{pmatrix} \dot{x} \\ \\ \dot{q} \end{pmatrix} + \Gamma_{fric}$$

This allows us to now isolate the $\dot{q}$ dynamics as follows:

$$\ddot{q} = \bar{A}_q(C' - Q) + \begin{pmatrix} \bar{a}_{xq} & \bar{A}_{qq} \end{pmatrix} \begin{pmatrix} {}^{-\tau_1}/_R \\ \Gamma \end{pmatrix} \tag{4.9}$$

$$\ddot{q} + \bar{A}_q(-C' + Q) = \mathcal{A}^{-1}\Gamma \tag{4.10}$$

where we have introduced the matrix $\mathcal{A}$ such that

$$\mathcal{A}^{-1}\Gamma = \begin{pmatrix} \bar{a}_{xq} & \bar{A}_{qq} \end{pmatrix} \begin{pmatrix} {}^{-\tau_1}/_R \\ \Gamma \end{pmatrix} = \Big( \begin{bmatrix} \frac{-\bar{a}_{xq}}{R} & 0_{n\times(n-1)} \end{bmatrix} + \bar{A}_{qq} \Big)\Gamma = \big(\bar{B} + \bar{A}_{qq}\big)\Gamma$$

$$\Rightarrow \mathcal{A}^{-1} = \bar{A}_{qq} + \bar{B} \tag{4.11}$$

where we have defined the matrix $\bar{B} = \begin{bmatrix} \frac{-\bar{a}_{xq}}{R} & 0_{n\times(n-1)} \end{bmatrix}$. Now left-multiplying eq. (4.10) with $\mathcal{A}$ gives us:

$$\mathcal{A}\ddot{q} + P(-C' + Q) = \Gamma \tag{4.12}$$

where $P = \mathcal{A}\bar{A}_q$. Eq. (4.12) represents the $\dot{q}$ dynamics on which whole-body control techniques are now applicable. $\mathcal{A}$ represents the new inertia matrix. Note that we only have the expression for its inverse so far. We can use that, and the fact that $rank(\bar{B}) = 1$, to derive expressions for $\mathcal{A}$ and $P$ as follows:

$$\begin{aligned} \mathcal{A} &= \Big(\bar{A}_{qq} + \bar{B}\Big)^{-1} \\ &= \bar{A}_{qq}^{-1} - \frac{1}{1+\alpha}\bar{A}_{qq}^{-1}\bar{B}\bar{A}_{qq}^{-1} \qquad \text{(ref. [67])} \\ &= (I - \beta B)A_{qq}^{*} \end{aligned} \tag{4.13}$$

$$
\begin{aligned}
P &= \mathcal{A}\bar{A}_q \\
&= (I - \beta B) A^*_{qq} \begin{pmatrix} \bar{a}_{xq} & \bar{A}_{qq} \end{pmatrix} \\
&= (I - \beta B) \begin{pmatrix} -\frac{a_{xq}}{a_{xx}} & I \end{pmatrix}
\end{aligned} \tag{4.14}
$$

where

$$
\begin{aligned}
\bar{A}^{-1}_{qq} &= A^*_{qq} \text{ defined in (4.3)} \\
B &= A^*_{qq}\bar{B} = \begin{bmatrix} -\frac{1}{R} A^*_{qq}\bar{a}_{xq} & 0_{n\times(n-1)} \end{bmatrix} \\
&= \begin{bmatrix} \frac{a_{xq}}{Ra_{xx}} & 0_{n\times(n-1)} \end{bmatrix} \text{ (using property (4.7))} \\
\beta &= \frac{1}{1+\alpha} \\
\alpha &= \mathrm{tr}(\bar{B}\bar{A}^{-1}_{qq}) = \mathrm{tr}(\bar{B}A^*_{qq}) = \mathrm{tr}(A^*_{qq}\bar{B}) = \mathrm{tr}(B) \\
&= \frac{a_{xq1}}{Ra_{xx}} \\
a_{xq1} &= \text{first element of } a_{xq}
\end{aligned}
$$

Note that we have derived the expressions for both matrices in terms of the elements of original inertia matrix $A$ so that there is no need for inverting the matrix to find the new matrices. Also, note that the new inertia matrix $\mathcal{A}$ is not symmetric due to the addition of the non-symmetric matrix $\beta B A^*_{qq}$ to an otherwise symmetric matrix $A^*_{qq}$. This asymmetry introduced into the inertia matrix of the new dynamics proves to be a limitation for multi-level task prioritization scheme in operational space control. We will illustrate this in the subsequent sections.

## 4.3 Operational Space Control And Task Prioritization

Given a fully actuated system of the form

$$
M\ddot{q} + h = \Gamma \tag{4.15}
$$

where $M$ is the inertia matrix and $h$ is the lumpsum effect of the Coriolis, gravitational and friction forces, we can control dynamics of a certain task in operational space $x_t$ by

defining the control law in terms of task-specific forces $F_t$ acting in operational space [25]. The method to deduce these forces is to replace $\Gamma$ with $J_t^T F_t$ and left-multiplying eq. (4.15) with $J_t M^{-1}$ resulting in the following operational space dynamics

$$\begin{aligned} \ddot{x}_t - \dot{J}_t\dot{q} + J_t M^{-1} h &= J_t M^{-1} J_t^T F_t \\ \Rightarrow F_t &= \Lambda_t \left( \ddot{x}_t - \dot{J}_t\dot{q} \right) + \bar{J}_t^T h \end{aligned} \tag{4.16}$$

where $J_t$ is the Jacobian of the task $x_t$, meaning $\dot{x}_t = J_t\dot{q}$ and $\ddot{x}_t = J_t\ddot{q} + \dot{J}_t\dot{q}$. And $\Lambda_t = \left(J_t M^{-1} J_t^T\right)^{-1}$ and $\bar{J}_t = M^{-1} J_t^T \Lambda_t$ are respectively the dynamically consistent task inertia matrix and task Jacobian pseudo-inverse.

Using $\ddot{x}_t^{ref}$ in place of $\ddot{x}_t$ in eq. (4.16), the resulting $F_t$ can be used in the following control law to give the linearized dynamics $\ddot{x}_t = \ddot{x}_t^{ref}$.

$$\Gamma = \underbrace{J_t^T F_t}_{\Gamma_t} + N_t^T \Gamma_o \tag{4.17}$$

where $N_t = I - \bar{J}_t J_t$ is the null-matrix of the Jacobian, due to the useful property of $J_t N_t = 0$, and $\Gamma_0$ is any arbitrary control that has no effect on the task dynamics $\ddot{x}_t$. This can be easily verified by left-multiplying (4.15)—using $\Gamma$ from (4.17)—with $J_t M^{-1}$. We see that $\Gamma_0$ vanishes, implying it has no effect of $\ddot{x}_t$. This then allows us to use a prioritization scheme [1, Chapter-3] such that we can subject a lower-priority task $x_l$ (such as posture control) to the constraints of a higher-priority task (such as balancing). This is achieved by using $\Gamma_0 = J_l^T F_{l|t}$ in (4.17), substituting the resulting $\Gamma$ in (4.15) and then left-multiplying (4.15) with $J_l M^{-1}$. This results in

$$\begin{aligned} \ddot{x}_l - \dot{J}_l\dot{q} + J_l M^{-1} h &= J_l M^{-1} \Gamma_t + J_l M^{-1} \underbrace{N_t^T J_l^T}_{J_{l|t}^T} F_{l|t} \\ \Rightarrow F_{l|t} &= \Lambda_{l|t} \left( \ddot{x}_l - \dot{J}_l\dot{q} \right) + \bar{J}_{l|t}^T \left( h - \Gamma_t \right) \end{aligned} \tag{4.18}$$

where

$$\begin{aligned}\Lambda_{l|t} &= (J_l M^{-1} J_{l|t}^T)^{-1} \\ &= (J_{l|t} M^{-1} J_{l|t}^T)^{-1}\end{aligned} \tag{4.19}$$

$$\begin{aligned}\bar{J}_{l|t} &= M^{-1} J_l^T (J_{l|t} M^{-1} J_l^T)^{-1} \\ &= M^{-1} J_{l|t}^T (J_{l|t} M^{-1} J_{l|t}^T)^{-1} \\ &= M^{-1} J_{l|t}^T \Lambda_{l|t}\end{aligned} \tag{4.20}$$

Here the original expressions obtained from the equation have been transformed to a form consistent with the forms of $\Lambda_t$ and $\bar{J}_t$ expressions by using the following property of the null-matrix [1, Chapter-3].

$$N_t M^{-1} = M^{-1} N_t^T = N_t M^{-1} N_t^T \tag{4.21}$$

Now using $\ddot{x}_l^{ref}$ in place of $\ddot{x}_l$ in (4.18), the resulting $F_{l|t}$ can be used in the following control law to obtain linearized system $\ddot{x}_l = \ddot{x}_l^{ref}$

$$\begin{aligned}\Gamma &= \Gamma_t + \Gamma_{l|t} \\ &= J_t^T F_t + J_{l|t}^T F_{l|t}\end{aligned} \tag{4.22}$$

If we want further prioritization, we will need to design that controller in the null-space of both tasks considered thus far. This is achieved using the following control law [1, Chapter-3]:

$$\Gamma = \Gamma_t + \Gamma_{l|t} + \underbrace{N_t^T N_{l|t}^T}_{\text{commutes}} \Gamma_o \tag{4.23}$$

where $N_{l|t} = I - \bar{J}_{l|t}J_{l|t}$ and $\Gamma_o$ is any arbitrary input neither affecting $\ddot{x}_t$ nor $\ddot{x}_l$ dynamics. Note that this works only because $N_t^T N_{l|t}^T$ possesses the important property of commutation. This helps us prove nullification by this product of both $J_t M^{-1}$ and $J_l M^{-1}$ hence nullifying the effects of lower priority controllers $\Gamma_o$. It is important also to note that the proof of this commutation again uses property (4.21).

### 4.3.1 Limitation Due To Asymmetric Inertia Matrix

Now coming back to the system of equations we developed in (4.12)

$$\mathcal{A}\ddot{q} + P(-C' + Q) = \Gamma$$

This system differs from the fully actuated system discussed in the last subsection in only one respect. The inertia matrix is not symmetric. If we follow the steps discussed in the preceding subsection to implement the null space control, we observe that the $x_t$ control (4.17) and $x_l$ control (4.22) (of one level of priority) will work by replacing $M$ with $\mathcal{A}$ and $h$ with $P(-C' + Q)$ wherever necessary, and the following definitions:

$$\begin{aligned}
\Lambda_t &= (J_t \mathcal{A}^{-1} J_t^T)^{-1} & \Lambda_t^T &= (J_t \mathcal{A}^{-T} J_t^T)^{-1} \\
\bar{J}_t &= \mathcal{A}^{-T} J_t^T \Lambda_t^T & \bar{J}_t^T &= \Lambda_t J_t \mathcal{A}^{-1} \\
\Lambda_{l|t} &= (J_l \mathcal{A}^{-1} J_{l|t}^T)^{-1} & \Lambda_{l|t} &\neq (J_{l|t} \mathcal{A}^{-1} J_{l|t}^T)^{-1} \\
\bar{J}_{l|t} &= \mathcal{A}^{-T} J_l^T (J_{l|t} \mathcal{A}^{-T} J_l^T)^{-1} & \bar{J}_{l|t} &\neq \mathcal{A}^{-1} J_{l|t}^T \Lambda_{l|t} \\
&= \mathcal{A}^{-T} J_l^T \Lambda_{l|t}^T
\end{aligned}$$

It is important to note that the dynamically consistent task inertia matrix is not symmetric and so its transpose has to be evaluated with the specific transposes as defined above. Also, the inertia matrix $\Lambda_{l|t}$ and pseudo-inverse $\bar{J}_{l|t}$ of the lower priority task can no longer be brought to forms similar to the forms presented in eqs. (4.19) as the property (4.21) on

which the specific transformation relied on, no longer holds. And this leads to the most important limitation of the lack of symmetry, which is, that prioritization fails at lower levels. Only one level of prioritization works. The control law (4.23) will not work for higher levels of prioritization because the commutation property of $N_t^T N_{l|t}^T$ no longer holds due again to its dependence on the validity of property (4.21), that is not there due to $\mathcal{A}$ being asymmetric.

### 4.3.2 Coordinate Transformation To Symmetric Inertia Matrix

We now present a coordinate transformation such that the inertia matrix of the transformed system is symmetric. More specifically, given the form of $\mathcal{A}$ we choose the new matrix to be $A_{qq}^*$. So we define a coordinate transformation $\dot{q}^* = J^* \dot{q}$ such that

$$\mathcal{A}\ddot{q} = A_{qq}^* \ddot{q}^* + S \tag{4.24}$$

Where $S$ is a residual term to be found. Given the definition above, we can determine expressions for $J^*$, $J^{*-1}$ and $S$ as follows. We have that,

$$\begin{aligned} \dot{q}^* &= J^* \dot{q} \\ \ddot{q}^* &= J^* \ddot{q} + \dot{J}^* \dot{q} \qquad\qquad \Rightarrow \ddot{q} = J^{*-1}\left(\ddot{q}^* - \dot{J}^* \dot{q}\right) \end{aligned} \tag{4.25}$$

Substituting in (4.24), we get

$$\mathcal{A} J^{*-1}\left(\ddot{q}^* - \dot{J}^* \dot{q}\right) = A_{qq}^* \ddot{q}^* + S \tag{4.26}$$

By comparing the terms on both sides, we get

$$\mathcal{A}J^{*-1} = A^*_{qq} \tag{4.27}$$

$$\begin{aligned} S &= -\mathcal{A}J^{*-1}\dot{J}^*\dot{q} \\ &= -A^*_{qq}\dot{J}^*\dot{q} \end{aligned} \tag{4.28}$$

Eq. (4.24) therefore becomes

$$\mathcal{A}\ddot{q} = A^*_{qq}\ddot{q}^* - A^*_{qq}\dot{J}^*\dot{q} \tag{4.29}$$

Using (4.27) we can find expressions for $J^*$ and $J^{*-1}$ as follows

$$\begin{aligned} J^{*-1} &= \mathcal{A}^{-1}A^*_{qq} & \qquad J^* &= A^{*-1}_{qq}\mathcal{A} \\ &= (\bar{A}_{qq} + \bar{B})A^*_{qq} & &= A^{*-1}_{qq}(I - \beta B)A^*_{qq} \\ &= I + \bar{B}A^*_{qq} & &= I - \beta A^{*-1}_{qq}BA^*_{qq} \end{aligned}$$

where we have utilized eqs. (4.3), (4.11) and (4.13) for simplifications. Finally, the original dynamical equation (4.12) becomes

$$A^*_{qq}\ddot{q}^* - A^*_{qq}\dot{J}^*\dot{q} + P(-C' + Q) = \Gamma \tag{4.30}$$

This equation represents the dynamics of the new coordinates $\ddot{q}^*$. The inertia matrix in this dynamic system is symmetric and we can therefore leverage the full potential of the operational space tools presented in preceding subsections.

If we wish to apply operational space control on this system, we will need task Jacobians with respect to the new coordinates $q^*$. Let's use $J^*_t$ to represent the new Jacobian for

task $x_t$

$$\dot{x}_t = J_t^* \dot{q}^*$$

$$\dot{x}_t = \underbrace{J_t^* J^*}_{J_t} \dot{q}$$

$$\Rightarrow J_t^* = J_t J^{*-1} \tag{4.31}$$

So, given task Jacobian with respect to the original coordinates, we can use $J^{*-1}$ to find the Jacobian with respect to the new coordinates using eq. (4.31).

## 4.4 Optimization Based Control

As mentioned in Chapter 2, using inverse dynamics (ID) and inverse kinematics (IK) as constraints for optimization programs minimizing a single-step cost set up to perform a desired task in the operational space, has gained popularity recently for whole-body control. We prefer the work of [2] because it offers flexibility in terms of incorporating constraints such as joint, position and torque limits, obstacle avoidance and allows prioritization among tasks. The minimization variables $X$ are joint torques, accelerations, speeds or contact wrenches. Dynamics are linear in joint accelerations. Similarly, kinematics are linear in joint speeds. The optimization program is set up as a quadratic cost as follows:

$$0.5 X^\top G X + g^\top X$$

$$\text{s.t } C_E X + c_E = 0$$

$$C_I X + c_I \leq 0$$

The cost function takes the form $0.5 \left\| PX - b \right\|^2$ so $G = P^\top P$ and $g = -P^\top b$. An example of cost function designed to perform task $x_t$ will have $X = \ddot{q}$, $P = J_t$, $b = \ddot{x}_t^* - \dot{J}_t \dot{q}$ and $\ddot{x}_t^* = K_p(x_t^{ref} - x_t) + K_d(\dot{x}_t^{ref} - \dot{x}_t) + \ddot{x}_t^{ref}$. The dynamics of manipulator (4.12) will be used as the equality constraint for solving the ID. This are quadratic programming problems

efficiently solved using many available software.

For prioritizing among multiple tasks, $P$ and $b$ can be re-written as [2]:

$$P = \begin{bmatrix} w_o P_o \\ w_1 P_1 \\ \vdots \\ w_n P_n \end{bmatrix}, b = \begin{bmatrix} w_o b_o \\ w_1 b_1 \\ \vdots \\ w_n b_n \end{bmatrix} \tag{4.32}$$

A set of desirable robot behaviors are specified via this cost function as per the goal of the high level controller with penalties for deviation of the robot from these desired behaviors [2].

## 4.5 Simulation Results

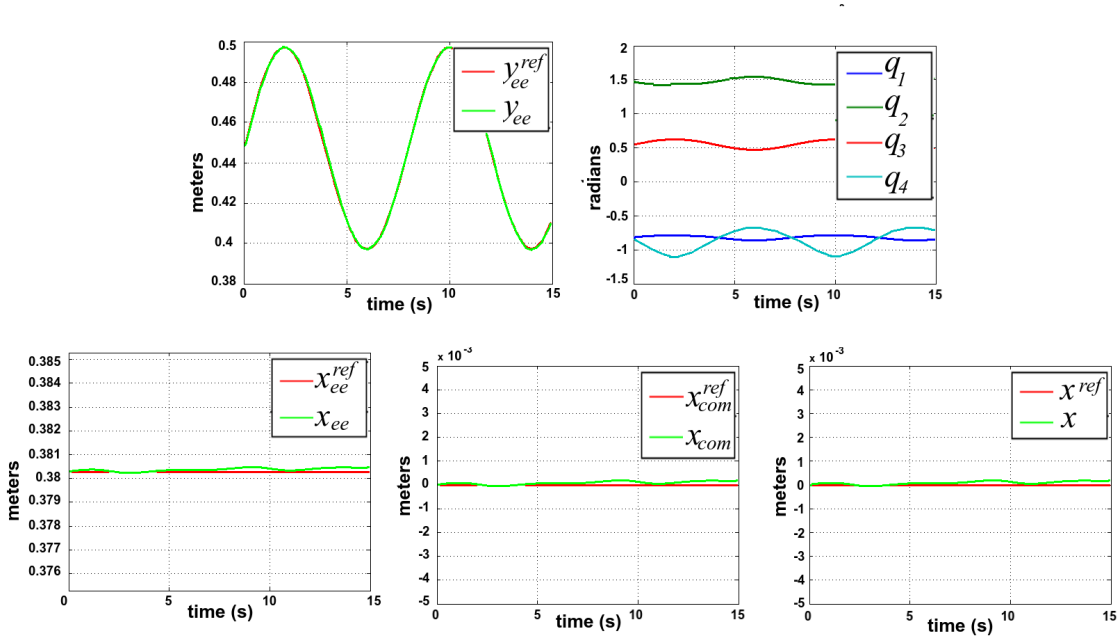



Figure 4.2: Plots of $y$-coordinate of the end-effector ($y_{ee}$), x-coordinate of the end-effector ($x_{ee}$), $x$-coordinate of the center of mass ($x_{com}$), horizontal position of the robot ($x$), and manipulator joint angles ($q_1, ..., q_4$)

We tested our method on a 5-DOF planar robot. The parameters used for the simulation

Table 4.1: Parameters used for simulation

| **Parameters** | **Symbols** | **Values** | **Units** | **Parameters** | **Symbols** | **Values** | **Units** |
|---|---|---|---|---|---|---|---|
| Wheel Radius | $R$ | 25 | $cm$ | Link center of masses | $c_1$ | 7.6 | $cm$ |
| Wheel mass | $m_w$ | 0.51 | $kg$ | (assumed to be along | $c_2$ | 8.1 | $cm$ |
| Wheel inertia | $J_w$ | $5.1 \times 10^{-3}$ | $kgm^2$ | the length) | $c_3$ | 15 | $cm$ |
| Link lengths | $L_1$ | 42.4 | $cm$ | | $c_4$ | 15 | $cm$ |
| | $L_2$ | 60 | $cm$ | Link viscous friction | $\mu_1$ | 15 | |
| | $L_3$ | 25 | $cm$ | coefficients | $\mu_2$ | 15 | |
| | $L_4$ | 25 | $cm$ | | $\mu_3$ | 15 | |
| Link masses | $m_1$ | 75.7 | $kg$ | | $\mu_4$ | 15 | |
| | $m_2$ | 14 | $kg$ | $x_{com}$ gains | $K_{pcom}$ | 0.1 | |
| | $m_3$ | 42 | $kg$ | | $K_{vcom}$ | 0.1 | |
| | $m_4$ | 14 | $kg$ | $r_{ee}$ gains | $K_{pee}$ | $100I_2$ | |
| Link inertias | $J_1$ | $75 \times 10^{-1}$ | $kgm^2$ | | $K_{vee}$ | $100I_2$ | |
| | $J_2$ | $14 \times 10^{-1}$ | $kgm^2$ | $x$ gains | $K_{px}$ | 0 | |
| | $J_3$ | $42 \times 10^{-1}$ | $kgm^2$ | | $K_{vx}$ | 10 | |
| | $J_4$ | $14 \times 10^{-1}$ | $kgm^2$ | | | | |

are listed down in Table 4.1. We controlled the motion of the end-effector to follow a trajectory while the robot maintains balance and the wheels stay at rest. We set the initial conditions to be $q(0^+) = \begin{bmatrix} -0.8157 & 1.4696 & 0.5505 & -0.8514 \end{bmatrix}$ radians, and the values of $x(0^+)$, $\dot{x}(0^+)$, $\dot{q}(0^+)$ were all set to zero. We controlled $x_{com}^{ref} = 0$ as the highest-priority task, end-effector position $r_{ee}^{ref} = [x_{ee}^{ref}, y_{ee}^{ref}]^T = [x_{ee}^{init}, y_{ee}^{init} + 5sin(2\pi t/8)]^T cm$ as the secondary task, horizontal position $x^{ref} = 0$ as the third priority task. Each linearized task input $\ddot{x}_t^{ref}$ was chosen to be:

$$\ddot{x}_t^{ref} = -K_{tp}(x_t - x_t^{ref}) - K_{tv}(\dot{x}_t - \dot{x}_t^{ref})$$

The desired speeds $\dot{x}_t^{ref}$ for each task were chosen to be zero, except for $y_{ee}^{ref}$ which was chosen to be the derivative of the sinusoidal reference trajectory. Figure 4.2 shows the result of simulation. We note that the end-effector follows the desired trajectory accurately as it maintains balance and its wheels stay at rest. The errors in $x_{ee}$, $x_{com}$ and $x$ are negligible. The last plot $(q_1, ..., q_4)$ shows that that all joints move, meaning the entire body participates in the task. A link to the simulation of the full robot can be found in [68].

## 4.6 Conclusion

We have presented a framework for whole-body control of a wheeled-inverted pendulum humanoid. Our primary contribution is to implement operational space control techniques on a class of under-actuated systems in which multiple degrees of freedom share the same actuator. We have considered the planar WIP robot dynamics in our analysis, but we believe the results can be extended to the 3D robot dynamics with a similar approach.

# CHAPTER 5

# HIGH-LEVEL CONTROLLER FOR CENTER OF MASS TRAJECTORY

In this chapter, we present a method to control the forward motion of WIP Humanoids. In the previous chapter, we isolated the manipulator/body dynamics from wheel dynamics and presented methods to control the former to achieve tasks such as end-effector motion and controlling body center of mass (CoM). As previously noted, wheel motor torque is also driving the base link. Full-body controller, proposed in the previous chapter, utilizes this torque to directly control only the upper body motions ignoring the wheel dynamics that result from it. This is the zero dynamics of the system. The high-level controller proposed in this chapter therefore control the wheels by planning for body motions. Specifically, since the dominant component in the zero dynamics is the torque due to body weight, we will plan for body center of mass to achieve the desired motion of the wheels.

Once the zero dynamics is derived, the problem reduces to one of determining the trajectory for body links that tracks a desired horizontal motion of the wheels. This can be set up as a simple trajectory optimization problem where a quadratic cost in terms of tracking error is to be minimized. But this control trajectory will be "open loop", having no "policy" in place if the system suffers deviation from optimal path due to disturbances or unmodeled dynamics. The robust version of this problem is referred as optimal control. Optimal control, not only gives us a control trajectory but also a feedback law that determines control sequence as a function of the current state. Hence, the system knows what to do at any given state to reach to the destination optimally. Dynamic Programming is a method that utilizes Bellman equation to find such a control policy. However, it suffers from what Bellman termed as "the curse of dimensionality". The storage requirements grow exponentially with the number of states, and thus approximate methods have to be utilized to solve for systems with more than a few states. Differential Dynamic Programming (DDP) [69] is a

solution to the curse of dimensionality, in that it approximates the value function (optimal cost as a function of initial state) as a second order function around the optimal trajectory and hence gives a control policy for deviation of the state. The storage requirement in this method grows only linearly with the states.

## 5.1 Simplified Model

The high-level planner we propose sees the robot as a simplified model that approximates the overall dynamics. This approach is similar to [2], where they approximate the bipedal humanoid with a Linear Inverted Pendulum Model (LIPM). We propose to use a Wheeled Inverted Pendulum Model (WIPM). Again, we will restrict our analysis to the planar case and leave its extension to 3D case as part of future work. The WIPM is a one-link robot dynamically balancing itself on the wheel (Figure 5.1). The results from Kane's analysis can help us quickly derive its model. We use $\theta$ to represent the pitch of the link. For the full robot, $\theta$ is the angular position of the body center of mass. Using Equations 3.6 and 3.9, the model of WIPM is:

$$\Big(m_w + \frac{I_w}{R^2}\Big)\ddot{x} + M\bar{a}_{G1}\bar{i}_o = \frac{-\tau_1}{R}$$

$$\Big(\bar{r}_{G1} \times M\bar{a}_{G1} + \mathbf{J}_{G1}\bar{\alpha}_1 + \bar{\omega}_1 \times (\mathbf{J}_{G1}\bar{\omega}_1)\Big) \cdot \bar{j}_o = \tau_1 + (\bar{r}_{G1} \times M\bar{g}) \cdot \bar{j}_o$$

Using $\bar{\omega}_1 = \dot{\theta}\bar{j}_o$, $\bar{\alpha}_1 = \ddot{\theta}\bar{j}_o$, $\bar{r}_{G1/O_o} = X_{com}\bar{i}_o + Z_{com}\bar{k}_o$, $\mathbf{J}_{G1} = I$ and $\bar{a}_{G1} = \bar{a}_o + (\bar{\alpha}_1 \times \bar{r}_{G1/O_o}) + \bar{\omega}_1 \times (\bar{\omega}_1 \times \bar{r}_{G1/O_o})$ $= \Big(\ddot{x} + Z_{com}\ddot{\theta} - X_{com}\dot{\theta}^2\Big)\bar{i}_o + \Big(-X_{com}\ddot{\theta} - Z_{com}\dot{\theta}^2\Big)\bar{k}_o$, the equations now become

$$R\Big(m_w + \frac{I_w}{R^2} + M\Big)\ddot{x} + RMZ_{com}\ddot{\theta} + RMX_{com}\dot{\theta}^2 = -\tau_1 \tag{5.1}$$

$$MZ_{com}\ddot{x} + (I + ML^2)\ddot{\theta} - MX_{com}g = \tau_1 \tag{5.2}$$

where we defined $L = |\bar{r}_{G1}| = \sqrt{X_{com}^2 + Z_{com}^2}$. Also $X_{com} = L\sin\theta$ and $Z_{com} = L\cos\theta$

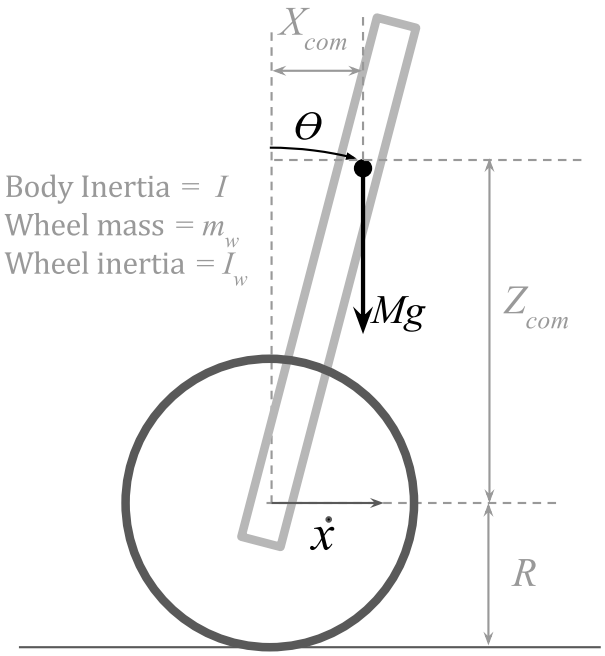



Figure 5.1: Wheeled Inverted Pendulum Model

## 5.2 Trajectory Planning on Simplified Model

The method in Chapter 4 can be used to control $\theta$ in the simplified model using $\tau_1$. What we are interested in here is to control the horizontal motion $\dot{x}$ using $\theta$. So we plan for a trajectory of $\theta$ in order to control $\dot{x}$. Eliminating $\tau_1$ from (5.1) and (5.2) gives us the zero dynamics of the system

$$\left[R\left(m+\frac{I_w}{R^2}+M\right)+MZ_{com}\right]\ddot{x}+\left(MZ_{com}+I+ML^2\right)\ddot{\theta}+RMX_{com}\dot{\theta}^2-MgX_{com}=0 \tag{5.3}$$

Dividing both sides of the equation with $ML$, we get:

$$(\alpha+\cos\theta)\,\ddot{x}+(\beta+\cos\theta)\,\ddot{\theta}=g\sin\theta-R\sin\theta\dot{\theta}^2 \tag{5.4}$$

where $\alpha=\frac{R}{ML}\left(M+m+\frac{I_w}{R^2}\right)$ and $\beta=\frac{I+ML^2}{ML}$.

Define $X = [\theta \quad \dot{\theta} \quad x \quad \dot{x}]^\top$ and $u = \ddot{\theta}$, we can rewrite the dynamics as

$$\dot{X} = \begin{bmatrix} \dot{\theta} \\ u \\ \dot{x} \\ \frac{g\sin\theta - R\sin\theta\dot{\theta}^2 - (\beta+\cos\theta)u}{\alpha+\cos\theta} \end{bmatrix} = f(X,u) \tag{5.5}$$

Given a desired trajectory for horizontal motion $x^{ref}(t)$ and $\dot{x}^{ref}(t)$, we define an objective function

$$V(X_0) = \int_0^{t_f} L(X,u)dt = \int_0^{t_f} (X(t) - X^{ref}(t))^\top G(X(t) - X^{ref}(t)) + gu^2(t)dt \tag{5.6}$$

where $X^{ref}(t) = [0 \quad 0 \quad x^{ref}(t) \quad \dot{x}^{ref}(t)]^\top$ and $G$, $g$ represent penalty weights. Then the Hamiltonian $H(X,u,V_X) = L(X,u) + \langle V_X, f(X,u)\rangle$. We solve for minimization of $V$ in a receding horizon fashion under the constraint (5.5) using DDP.

### 5.2.1 Computational Procedure

DDP is an iterative procedure requiring an initial control trajectory $\bar{u}(t)$. We utilize LQR on a linearized model of (5.5) to obtain $\bar{u}(t)$. The iterations are as follows [69]:

- Use $\overline{u}(t)$ in (5.5) to get corresponding state trajectory $\overline{X}(t)$ cost $\overline{V}(X_0)$
- **Backward Pass:** Using boundary conditions $a(t_f) = 0$, $V_X(t_f) = 0$ and $V_{XX}(t_f) = 0$, integrate backward in time

$$\begin{aligned} -\dot{a} &= H - H(\overline{X}, \overline{u}, V_X) \\ -\dot{V}_X &= H_X + K^\top H_u + V_{XX}\left(f - f(\overline{X}, \overline{u})\right) \\ -\dot{V}_{XX} &= H_{XX} + f_X^\top V_{XX} + V_{XX} f_X - (H_{uX} + f_u^\top V_{XX})^\top H_{uu}^{-1}(H_{uX} + f_u^\top V_{XX}) \\ &\quad + \frac{1}{2}V_{XX}(f - f(\overline{X}, \overline{u})) + \frac{1}{2}(f - f(\overline{X}, \overline{u}))^\top V_{XX} \end{aligned} \tag{5.7}$$

all quantities are evaluated at $\bar{X}$ and $u^*$ unless otherwise stated. $u^*$ and $K$ are obtained by minimizing $H$ wrt $u$ at every step as follows

$$u^* = \overline{u} + \delta u^* \tag{5.8}$$

$$\delta u^* = \underset{\delta u}{\operatorname{argmin}}\, H(\overline{X}, \overline{u} + \delta u, V_X) \tag{5.9}$$

$$K = -H_{uu}^{-1}(H_{uX} + f_u^\top V_{XX}) \tag{5.10}$$

- **Forward Pass:** Apply control $u(t) = u^*(t) + K(t)\delta X(t)$ where $\delta X(t)$ is obtained by integrating:

$$\left(\frac{d}{dt}\right)(\overline{X} + \delta X) = f(\overline{X} + \delta X, u^* + K\delta X)$$
$$\overline{X}(t_o) + \delta X(t_o) = X_o \Rightarrow \delta X(t_o) = 0$$

  Resulting in a new trajectory $X(t)$ and corresponding cost $V(X_0)$

- Repeat the process until convergence of cost

The process results in an optimal control trajectory $u^*(t)$ and state trajectory $X^*(t)$. The control policy to be applied is

$$u(t) = u^*(t) + K(t)(X(t) - X^*(t))$$

### 5.3 Zero Dynamics of the Full Robot

We have laid down the method to plan trajectory for the approximated model. Now we investigate how the effect of unmodeled dynamics can be minimized. For that we derive here the zero dynamics of the full model. We find that by eliminating $\tau_1$ from the $\dot{x}$ and $\dot{q}_1$

equations in the dynamic model. The two equations of the dynamic model are

$$\begin{pmatrix} a_{xx} & a_{xq}^{\top} \\ a_{xq} & a_{qq_1}^{\top} \end{pmatrix} \begin{pmatrix} \ddot{x} \\ \ddot{q} \end{pmatrix} - \begin{pmatrix} C'_x \\ C'_{q_1} \end{pmatrix} + \begin{pmatrix} Q_x \\ Q_{q_1} \end{pmatrix} = \begin{pmatrix} -\frac{\tau_1}{R} \\ \tau_1 \end{pmatrix} \tag{5.11}$$

where $a_{qq_1}$ is the first row of $A_{qq}$, $\begin{bmatrix} C'_x & C'_{q_1} \end{bmatrix}^{\top}$ and $\begin{bmatrix} Q_x & Q_{q_1} \end{bmatrix}^{\top}$ are the first two elements of $C'$ and $Q$ respectively. Eliminating $\tau_1$ from the two equations results in

$$\begin{aligned} & (Ra_{xx} + a_{xq_1})\,\ddot{x} \quad + \quad \left(Ra_{xq}^{\top} + a_{qq_1}^{\top}\right)\ddot{q} \\ & \qquad\qquad - \left(RC'_x + C'_{q_1}\right) + (RQ_x + Q_{q_1}) = 0 \\ \Rightarrow \quad & \left[R\left(2m_w + \frac{2\mathbb{Y}_w}{R^2} + M\right) + MZ_{com}\right]\ddot{x} \quad + \quad \left[R\begin{pmatrix} MZ_{com} \\ M_2 Z_{com(2)} \\ \vdots \\ M_n Z_{com(n)} \end{pmatrix} + \begin{pmatrix} I + ML^2 \\ I_2 + \beta_2 \\ \vdots \\ I_n + \beta_n \end{pmatrix}\right]\ddot{q} \\ & \qquad\qquad - \left(RC'_x + C'_{q_1}\right) - MgX_{com} = 0 \end{aligned} \tag{5.12}$$

where

$M$ is full body mass
$I$ is full body inertia around wheel axis
$X_{com}, Z_{com}$ are coordinates of body CoM
$L = \sqrt{X_{com^2} + Z_{com}^2}$ is CoM distance from wheel axis
$M_j$ is the sum of masses of all links antecedent to link $j-1$
$Z_{com(j)}$ is the CoM of the system of links antecedent to link $j-1$ expressed in frame $R_0$
$I_j$ is the sum of inertia of all links antecedent to link $j-1$ about their respective CoMs
$\beta_j = \sum_{i=j}^{n} m_i \bar{r}_{Gi/O_j} \cdot \bar{r}_{Gi/O_1}$ are terms when added to $I_j$ evaluates to inertia about $O_0$ (parallel axis theorem)

Note that in deriving (3.50), we have utilized expressions for $a_{xx}$, $a_{xq_1}$,$a_{xq}$, $a_{qq_1}$, $Q_x$ and $Q_{q_1}$ determined from the Kane's equations and ignoring the third dimension. Note also that if quasi-static motions are assumed for upper body ($\ddot{q} = \dot{q} = 0$) then the only term driving $\dot{x}$ motion is the torque due to body weight $MgX_{com}$.

Zero dynamics of the simplified model are given in (3.56). Comparing with (3.50), we note that the two models have a one-to-one correspondence. The difference arises only because of $\dot{\theta}$ dynamics approximating the dynamics of the entire upper body. To analyse the effect of this approximation we derive expression of $\ddot{\theta}$ in terms of $\ddot{q}$.

$$\begin{aligned}
\theta &= \arctan \frac{X_{com}(q)}{Z_{com}(q)} \\
\dot{\theta} &= \underbrace{\frac{1}{L^2(q)} \left[ Z_{com}(q) \frac{\partial X_{com}(q)}{\partial q} - X_{com}(q) \frac{\partial Z_{com}(q)}{\partial q} \right]}_{J_\theta} \dot{q} \\
\ddot{\theta} &= \underbrace{\frac{1}{L^2(q)} \begin{pmatrix} Z_{com}^2(q) + X_{com}^2(q) \\ Z_{com} Z_{com(2)} + X_{com} X_{com(2)} \\ \vdots \\ Z_{com} Z_{com(n)} + X_{com} X_{com(n)} \end{pmatrix}^\top}_{J_\theta} \begin{pmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \vdots \\ \ddot{q}_n \end{pmatrix} + \dot{J}_\theta \dot{q}
\end{aligned}$$

where we used $\frac{\partial}{\partial q} X_{com}(q) = \begin{pmatrix} Z_{com}(q) & Z_{com(2)}(q) & \cdots & Z_{com(n)}(q) \end{pmatrix}$ and $\frac{\partial}{\partial q} Z_{com}(q) = \begin{pmatrix} -X_{com}(q) & -X_{com(2)}(q) & \cdots & -X_{com(n)}(q) \end{pmatrix}$.

Ignoring the terms involving squares of generalized speeds ($C'_x$, $C'_{q_1}$, $\dot{J}_\theta \dot{q}$ and $R X_{com} \dot{\theta}^2$) from the the zero dynamics of both models, the difference between the two dynamics is captured by the following expression:

$$\sum_{i=2}^{n} \ddot{q}_i \left( R M_i Z_{com(i)} + I_i + \beta_i - \frac{R M Z_{com} \left( I + M L^2 \right)}{L^2} \left( Z_{com} Z_{com(i)} + X_{com} X_{com(i)} \right) \right) \tag{5.13}$$

Note that the sum only involves $\ddot{q}_j$ for $j = 2 \cdots n$. This is because the inertial effects around joint $1$ are perfectly modeled by the simplified model and cancel out when evaluating the difference. These effects are only as large as the accelerations of individual joints, and

their effect is less significant compared to the effect of the torque due to body weight. The latter is perfectly modeled by the approximate model. In order to minimize the effect of these unmodeled dynamics in the DDP, we can add a constraint in the quadratic program for upper body control to keep the expression (5.13) equal to zero. This will ensure that the optimal control trajectory determined by DDP for the approximate model remains valid for the full model.

## 5.4 Receding Horizon Control

To close the loop for dealing with deviations from optimal path due unmodeled dynamics, we perform model predictive control (MPC) at sampling intervals $T_s$ for a short horizon $t_H = NT_s$. MPC performs trajectory optimization on a continuous basis over $t_H$ on an updated version of the simplified model (using the most recent $\theta$ dynamics and body moments), and its first control value is used as the reference for full-body control QPs over the next sampling period $T_s$.

We use differential dynamic programming (DDP) for trajectory optimization over the full horizon, as well as, optimization in every MPC iteration over the smaller horizon $t_H$. See [69] for the historical presentation and [70] for a modern treatment of DDP. It was developed as an approximate dynamic programming method to deal with the curse of dimensionality of the original dynamic programming approach proposed by Bellman. It works by iterating a forward pass on the discrete dynamics, followed by a backward pass which compute a local solution to the optimization problem using a quadratic Taylor expansion. Due to its fast convergence owing to second-order expansion of the value function, this method is gaining popularity in robotic applications.

Assuming small time-step $\Delta t$ the discretized dynamics of the system are

$$X_{i+1} = X_i + \Delta t f_c(X_i, u_i) = f(X_i, u_i).$$

Given a desired goal position $X^{ref} = \begin{bmatrix} \theta^{ref} & \dot{\theta}^{ref} & x^{ref} & \dot{x}^{ref} \end{bmatrix}^\top = \begin{bmatrix} 0 & 0 & x^{ref} & 0 \end{bmatrix}^\top$ for a given final time $t_f$, define a one step cost

$$L_i^{ddp} = (X_i - X^{ref})^\top G_{ddp}(X_i - X^{ref}) + g_{ddp} u_i^2,$$

where $G_{ddp}$ and $g_{ddp}$ are weights penalizing state deviation and control effort respectively. Then DDP can be used to generate a reference trajectory . This trajectory will be generated over the full horizon $t_f$ (or $N_{traj} = t_f/\Delta t$ steps), using the simplified model parameters $\lambda(q)$ at the initial state $q(0)$. Once we have the trajectory,

$$X^{traj}(i) = \begin{bmatrix} \theta^{traj}(i) & \dot{\theta}^{traj}(i) & x^{traj}(i) & \dot{x}^{traj}(i) \end{bmatrix}^\top$$

, it will be used as a time-varying reference for a receding horizong controller (or model predictive controller MPC) with a smaller horizon $t_H$ (or steps $N = t_H/\Delta t$) to generate closed loop control. Here the parameters of the simplified model $\lambda(q)$ will be updated at every time-step using the current state of the full robot. Define the one-step cost $L_i^{mpc} = (X_i - X_i^{traj})^\top G_{mpc}(X_i - X_i^{traj}) + g_{mpc} u_i^2$, the MPC scheme will generate a control sequence for the horizon $t_H$ by minimizing the cost over the horizon $t_H$. The first step of this control trajectory ($\ddot{\theta}^{ref}$) will be used as reference target for the low-level controller. The low-level controller will determine torques of all joints by minimizing a single step cost using quadratic programming as described earlier. This enables full-body participation in locomotion along with performing other manipulation tasks, such as end-effecotr orientation or gaze control.

### 5.5 Algorithm

Algorithm 1 summarizes the whole-body control scheme in algorithmic form.

**Algorithm 1** Algorithmic description of the hierarchical whole-body control scheme

**Input:** full horizon $t_f$, receding horizon $t_H << t_f$
Initial state of the simplified model $X(0) = \mathcal{O}_{4\times1}$,
Goal state of the simplified model $X(t_f) = \begin{bmatrix} 0 & 0 & x^{ref} & 0 \end{bmatrix}^\top$,
Set of other task-space objectives to be fulfilled $x_{tasks}^{ref}$ (*e.g.* end-effector position/orientation control, regulation *etc.*)

**Output:** Body torques $\Gamma(t)$ for each time-step through the full horizon

1: $N_{traj} \leftarrow t_f/\Delta t$
2: $N \leftarrow t_H/\Delta t$
3: Generate reference trajectory $X^{traj} = \{X_i^{traj}\}$, $u^{traj} = \{u_i^{traj}\}$, $i = 1, ..., N_{traj}$ using DDP

$$\{X^{traj}, u^{traj}\} \leftarrow \underset{u,X}{\operatorname{argmin}} \sum_{i=1}^{N_{traj}} L_i^{ddp}(X_i, u_i)$$
$$\text{s.t.} \quad X_{i+1} = f(X_i, u_i)$$

4: **for** $i = 1$ to $N_{traj}$ **do**
5: Update pose-dependent model parameters $\lambda(q)$ of the simplified model $f(.)$
6: Find trajectories over the short horizon $X^{ref} = \{X_j^{ref}\}$, $u^{ref} = \{u_j^{ref}\}$, $j = i, ..., i+N$

$$\{X^{ref}, u^{ref}\} \leftarrow \underset{u,X}{\operatorname{argmin}} \sum_{j=i}^{i+N} L_j^{mpc}(X_j, u_j)$$
$$\text{s.t.} \quad X_{j+1} = f(X_j, u_j)$$

7: Use first control value $u^{ref}(i) = \ddot{\theta}^{ref}$ as reference input for full-body controller (FBC) executed during the next sampling period $\Delta t$ that will compute all joint torques by solving QPs

$$\Gamma(t) \leftarrow \text{FBC}(\theta^{ref}, x_{tasks}^{ref})$$

8: **end for**

## 5.6 Conclusion

In this chapter we proposed a high-level controller that generates a target trajectory for the Center of Mass to be tracked by the full-body controller presented in the previous chapter in order to control the horizontal motion of the robot. In order to deal with the curse of dimensionality in dynamic programming we planned our motion for a model that approximates the full robot as a one-link wheeled inverted pendulum. This will ensure that the computation is fast if implemented online in a receding horizon fashion. We also developed closed form expressions for the effects of unmodeled dynamics. These effects can thus be minimized by augmenting the unmodeled effects as constraints in the quadratic programs solving for full-body control.

# CHAPTER 6

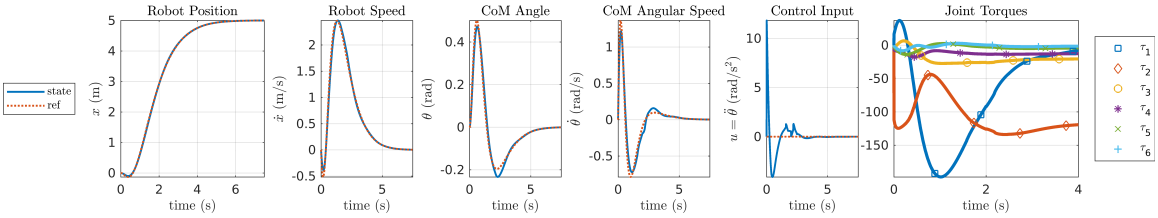
# IMPLEMENTATION AND RESULTS

We applied the presented techniques in three stages. Firstly, a 7-DOF planar WIP robot was modeled using MATLAB and the proposed techniques were implemented in MATLAB to control that model. Secondly, in order to demonstrate the approach on the simulation of the full model, we used off-the-shelf simulation tool, DART, which uses state-of-the-art dynamics algorithm based on Featherstone's techniques to model dynamics of highly articulated structures. Finally, we re-implemented the logic for final deployment on the hardware using for the software architecture described in the previous chapter.

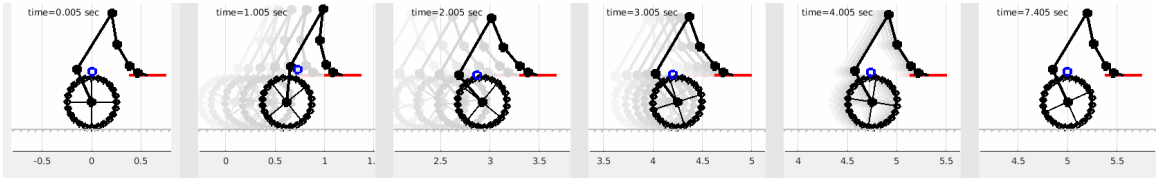
## 6.1 Planar Robot

We applied the presented approach on a 7-DOF planar robot *i.e.*, a six-link serial structure with wheels. Parameters used for the simulation are listed in supplementary material. These parameters are chosen to approximate the physical parameters of system in consideration (see Figure 3.1). Exception is friction parameters where we chose to omit the coloumb component and used only an arbitrary coefficient for kinetic friction. The task performed is to move the robot to a target location, $x = 5$ meters, while maintaining the position and orientation of the end-effector. Initial values for the state are $\begin{bmatrix} x(0) & q(0)^\top & \dot{x}(0) & \dot{q}(0)^\top \end{bmatrix}^\top = \begin{bmatrix} 0 & -18.6^\circ & 56.2^\circ & 138.7^\circ & -21.2^\circ & -21.2^\circ & -21.2^\circ & \mathcal{O}_{1\times 7} \end{bmatrix}^\top$. Then, the initial state for the simplified model $\begin{bmatrix} \theta(0) & \dot{\theta}(0) & x(0) & \dot{x}(0) \end{bmatrix}^\top = \mathcal{O}_{4\times 1}$.

For trajectory generation using DDP, typically, a small penalty is assigned for each step and a large penalty is assigned for the terminal step. We have used a similar scheme. In our experiments, it turns out that for robots with large masses, a higher weight needs to be assigned to the pendulum angle $\theta(t)$. This ensures that the trajectory generated remains




(a) Simplified model reference and state trajectories (first 5 plots). The last plot shows the resulting joint torques for $t \in (0, 4)$ seconds.




(b) Snapshot of the robot at various time instances during execution

Figure 6.1: Simulation results on a planar 7-DOF WIP humanoid

well within the stable region thus ensuring that the closed loop control (MPC) remains stable during execution. Note that MPC will not follow the trajectory generated by DDP exactly owing to the disturbances caused by full robot dynamics that were ignored in DDP. This means that $\theta$ may overshoot beyond the reference trajectory generated by DDP during execution. If $\theta$ was barely held within stable region, this may lead the full model to go unstable.

For MPC, a horizon of 1 sec is used for each optimization step. A higher step cost is assigned to positions ($x(t)$ and $\theta(t)$) compared to speeds ($\dot{x}(t)$ and $\dot{\theta}(t)$). This ensures that there is no error is final position of the robot and that pendulum angle stays within stable region. It is not as critical to accurately follow the reference speeds. In order to prevent the robot from overshooting in position *i.e.*, going beyond the target $5$ meters, we have assigned a very high weight ($10^7$) if $x > 5$ both in DDP and MPC. We have also used terminal weights for the MPC scheme, as they provide better tracking and stability performance. Terminal weights assign a high cost to the deviation of state at the end of the horizon at each control iteration. Final time $t_f = 20$ sec is used in the DDP to generate the results shown here. However the task was completed in $7.5$ sec.

Finally, for the full-body controller, we have the following $P$ and $b$ matrices

$$P = \begin{bmatrix} w_\theta J_\theta \\ w_{r_{ee}} J_{r_{ee}} \\ w_{\phi_{ee}} J_{\phi_{ee}} \\ w_{reg} \mathcal{I}_6 \end{bmatrix} \quad b = \begin{bmatrix} -w_\theta (\dot{J}_\theta \dot{q} - \ddot{\theta}^{ref}) \\ -w_{r_{ee}} (\dot{J}_{r_{ee}} \dot{q} - \ddot{r}_{ee}^{ref}) \\ -w_{\phi_{ee}} (\dot{J}_{\phi_{ee}} \dot{q} - \ddot{\phi}_{ee}^{ref}) \\ w_{reg} \ddot{q}_{reg} \end{bmatrix} \tag{6.1}$$

For each task $x_t \in \{\theta, r_{ee}, \phi_{ee}\}$ the reference acceleration $\ddot{x}_t^{ref}$ appears in the defintion of $b$ and is defined as

$$\ddot{x}_t^{ref} = \ddot{x}^d - K_p(x_t - x_t^d) - K_v(\dot{x}_t - \dot{x}_t^d)$$

The values of weights $w$ and PD gains ($K_p$ and $K_v$) used for the simulation are listed in the supplementary material. Desired position and orientation of the end-effector ($r_{ee}^d$ and $\phi_{ee}^d$) were set to their values at initial time. And the desired speeds and accelerations were set to zero. For pendulum angle $\theta$, the desired position and speed ($\theta^d$ and $\dot{\theta}^d$) come from the trajectory generated by DDP, while the desired acceleration $\ddot{\theta}^d$ is the control input determined by the MPC iteration. These reference values from DDP/MPC are used for the remaining sampling period ($0.01$ sec). During this period ode45 is used with full-body control QPs to simulate the behavior of the full robot.

Figure 6.3 shows the reference trajectory determined by DDP on the simplified model, and the state of the simplified model that results by applying the MPC control values on the full model. We see that the state follows the reference trajectory very closely with disturbances occuring during fast transitions owing to the disturbances caused by unmodeled full body dynamics. Also the last plot to the right is the plot of all joint torques. It is clear that the entire body is participating in performing the three tasks.

Snapshots of the full body at 8 different instances during execution are shown in figure 6.1b. For each snapshot, the pose of the body during five recent instances are also shown using shaded lines, to visualize the speeds of the bodies. The blue circle represents the

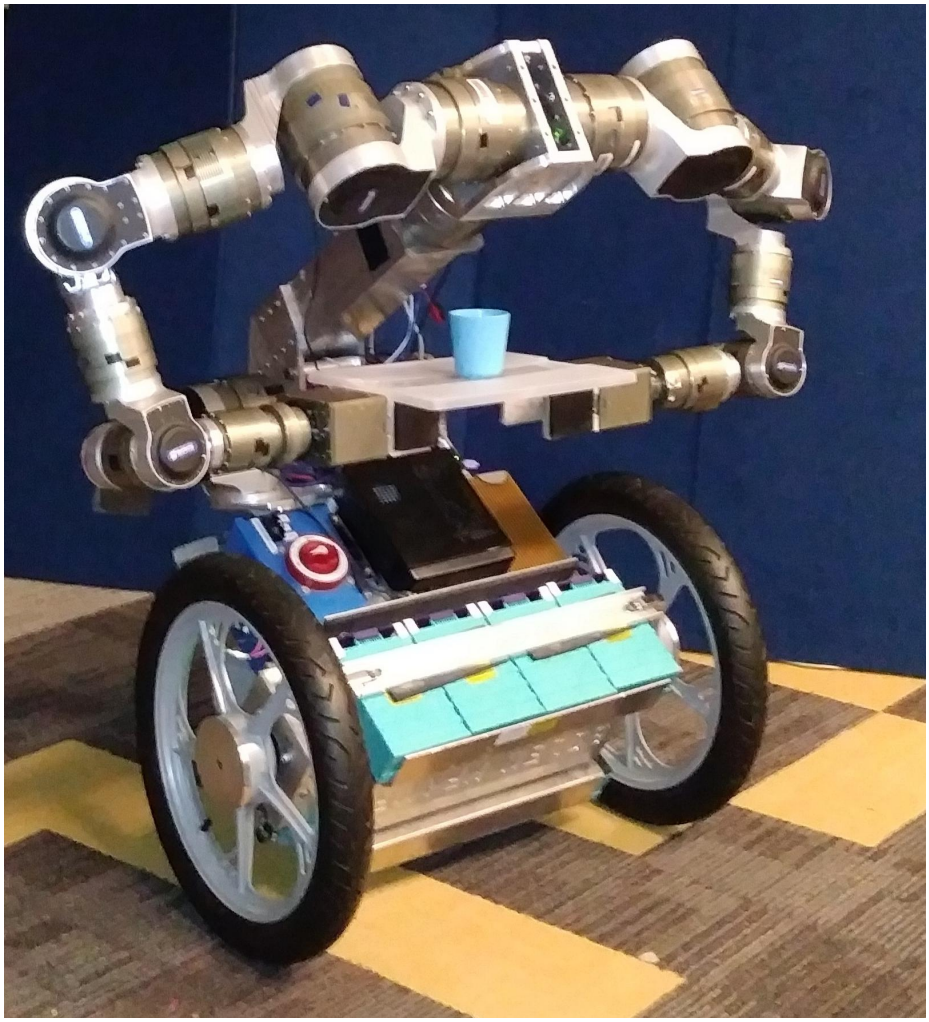


Figure 6.2: Krang in its balanced pose carrying a cup on a tray

CoM of the robot. And the red line represents an object that is attached to the last link at a fixed orientation. We see that the whole body is participating in manipulating the CoM to fulfill position objectives, while maintaining the orientation and position of the object attached to the end-effector.

## 6.2 3D Simulation of the Full Robot

The platform, Golem Krang (Fig 6.2), is equipped with an on-board computer, and precise position encoders for all joints. Real-time sensing and control is possible with many dedicated CAN-based physical interfaces, and the real-time operating system installed on the

Table 6.1: Simulation Parameters

| Parameters | Values | Units |
|---|---|---|
| ***Dynamic Model Parameters:*** | | |
| Wheel mass | 0.51 | kg |
| Wheel radius | 25 | cm |
| Wheel inertia | 0.0051 | kg-m$^2$ |
| Link lengths | 36, 66, 31, 25, 10, 10 | cm |
| Link masses | 75, 24, 6.3, 3.8, 2.6, 2.3 | kg |
| Link inertias | 2.5, 2.7, 0.16, 0.062, 0.0073, 0.0063 | kg-m$^2$ |
| Link CoMs (assumed along link length) | 18, 33, 16, 13, 5.2 , 5.2 | cm |
| Link viscous friction coefficients | 15, 15, 15, 15, 15, 15 | |
| ***High-Level Controller Parameters:*** | | |
| Sampling Time for DDP/MPC | 0.01 | sec |
| MPC Horizon | 1 | sec |
| $G_{ddp}$, Terminal $G_{ddp}$ | diag($\begin{bmatrix}0.1 & 1 & 0.1 & 0.1\end{bmatrix}$), | |
| $g_{ddp}$ | $10^4\mathcal{I}_4$, 0.01 | |
| $G_{mpc}$, Terminal $G_{mpc}$ | diag($\begin{bmatrix}10 & 10 & 0.1 & 0.1\end{bmatrix}$), | |
| and $g_{mpc}$ | $10^4$diag($\begin{bmatrix}10 & 10 & 0.1 & 0.1\end{bmatrix}$),0.01 | |
| ***Full-Body Controller Parameters:*** | | |
| $\theta$ Control (weight $w$, Position Gain $K_p$, Speed Gain $K_v$) | 10, 10, 10 | |
| End effector Position Control ($w$, $K_p$, $K_v$) | 1, $100\mathcal{I}_2$, $100\mathcal{I}_2$ | |
| End effector Orientation Control ($w$, $K_p$, $K_v$) | 1, 100, 100 | |
| Regulator ($w$, $K_v$) | 0.1, 10 | |

computer. For inter-process communication (IPC) among sensing/control and application processes with real-time guarantees, ACH [71] is used as the underlying framework. The lower body of Krang is very heavy ($\sim 100$ kg), making it suitable to perform dynamic tasks by manipulating its weight torque intelligently. These characteristics make this platform especially suited for trying out the various ideas for dynamic control of complex systems discussed earlier. This section is divided into two subsections. We first discuss results from the simulation-based investigations of model-based control that are geared specifically to overcome various challenges in implementing model-based hierarchical control on this platform. Secondly, we discuss our result on semi-parametric learning and control of a 7DOF arm of this robot in simulation.

### 6.2.1 Model-Based Hierarchical Control

A verification of the hierarchical control approach for whole-body control of Krang was done using DART simulation of Krang. Parameters for the model were chosen to reflect properties of the physical system. A simulation video is provided in the supplementary material [1].

Figure 6.3 shows the reference trajectory determined by DDP on the simplified model, and the state of the simplified model that results by applying the receding horizon control in a closed-loop manner on the simplified model. The control and state trajectories produced by this high-level controller are used as references for the low-level controller, which is also responsible for controlling the position and orientation of the end-effector. We see that the state follows the reference trajectory very closely with disturbances occurring during fast transitions owing to the disturbances caused by unmodeled full body dynamics (i.e. not modeled by the simplified model used by the high-level controller). Also, the last plot to the right is the plot of six joint torques from the full body. It is clear that the entire body is participating in performing the three tasks.

[1] Visit https://vimeo.com/user90167025/review/292912849/f662c39b8f

Snapshots of the full body at 4 instances during execution are shown in Figure 6.4. Fig 6.4A shows the perspective and side views of the case when we perform ID based low-level control for the full body. This control assumes perfect knowledge of model parameters. And it shows that the hierarchical controller is able to perform unified control of locomotion and manipulation. This is demonstrated by the cup on the tray carried by the arms. If we turn off whole-body control and perform simple locomotion task on the simplified model, the cup ends up falling. This is due to the changes in pitch of the body affecting the orientation of the end-effector. This however is prevented when the control is unified as shown in the figure.

In order to perform this experiment on the hardware, we needed to verify two more cases in simulation. The waist joint on the robot supports the bulk of weight of the upper body, and should not be unlocked after reaching the desired height. We investigated in simulation if it is necessary to unlock this joint to perform the tray-carrying task. We achieved control of simulated robot with a locked waist joint, by (a) excluding waist joint acceleration in the vector of decision variables for QP-based optimization in the low-level controller, and (b) deleting the columns and rows in the various Jacobian-based cost functions [72], and EOMs for ID that correspond to the waist joint. Fig 6.4B shows the results. We see that when the waist is unable to lift the upper body to satisfy unified objectives, the base joint has to tilt the body a lot more to ensure CoM targets are being followed as the arms are busy in maintaining tray orientation.

A third point of investigation before hardware experiments could begin is the use of IK for control of the arms. This allows us to relax the assumption of perfect knowledge of system parameters, as kinematic references are followed by joint level PID controllers. Fig 6.4C shows the result. Again the waist joint is locked in this formulation. A key point to note for IK is that the robot joints are all harmonic drives with the effect of frictions much more significant than other dynamic effects. We reflected this fact in choosing our simulation frictions. In our experience, IK is a lot easier to perform in the presence of high

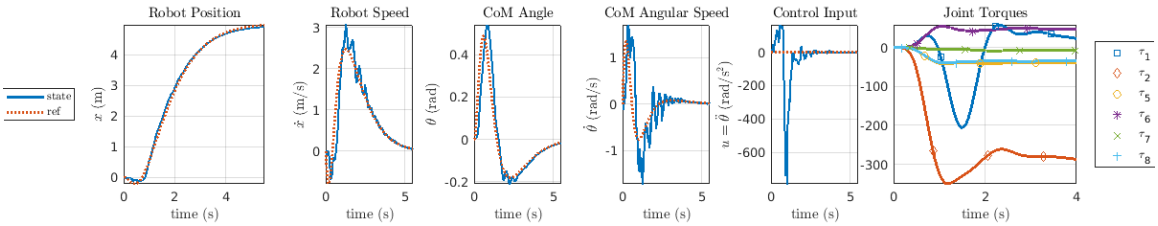



Figure 6.3: Simplified model reference and state trajectories (first 5 plots). The last plot shows the resulting joint torques for $t \in (0, 4)$ seconds.

viscous frictions, as the frictions are helpful in stabilizing the system around any configuration. This however is not the case for base, waist and torso joints. This is due to the inertial and gravitational effects of the articulated structures mounted on these three joints far outweigh the respective frictions on these joints. Therefore the use of ID-based control for these joints is indispensable. In the absence of perfect system knowledge, learning based approaches discussed above have to be utilized.

### 6.2.2 Semi-Parametric Learning On Krang's Arm

To verify the semi-parametric based approach on low-level controller, we first implement it on one of the serial manipulators on Krang. This is a necessary investigation before deploying the framework on the whole system. Non-linear error terms were included in the dynamics with the objective of GP capturing such dynamics. We successfully implemented it and verified it in simulation.

The experimental pipeline in performing the semi-parametric based approach incorporated the following steps: follow reference trajectory in operation space and collect training/testing datasets, compute hyper-parameters from training set, and perform online control.

The point cloud of end-effector positions are shown in Figure 6.5. Hyper-parameters were determined based on minimizing the log-marginal likelihood and online control is performed on test trajectory. The computation time for performing online control was of the order $5 \sim 12$ milliseconds per query point. The performance of GP-based torque com-

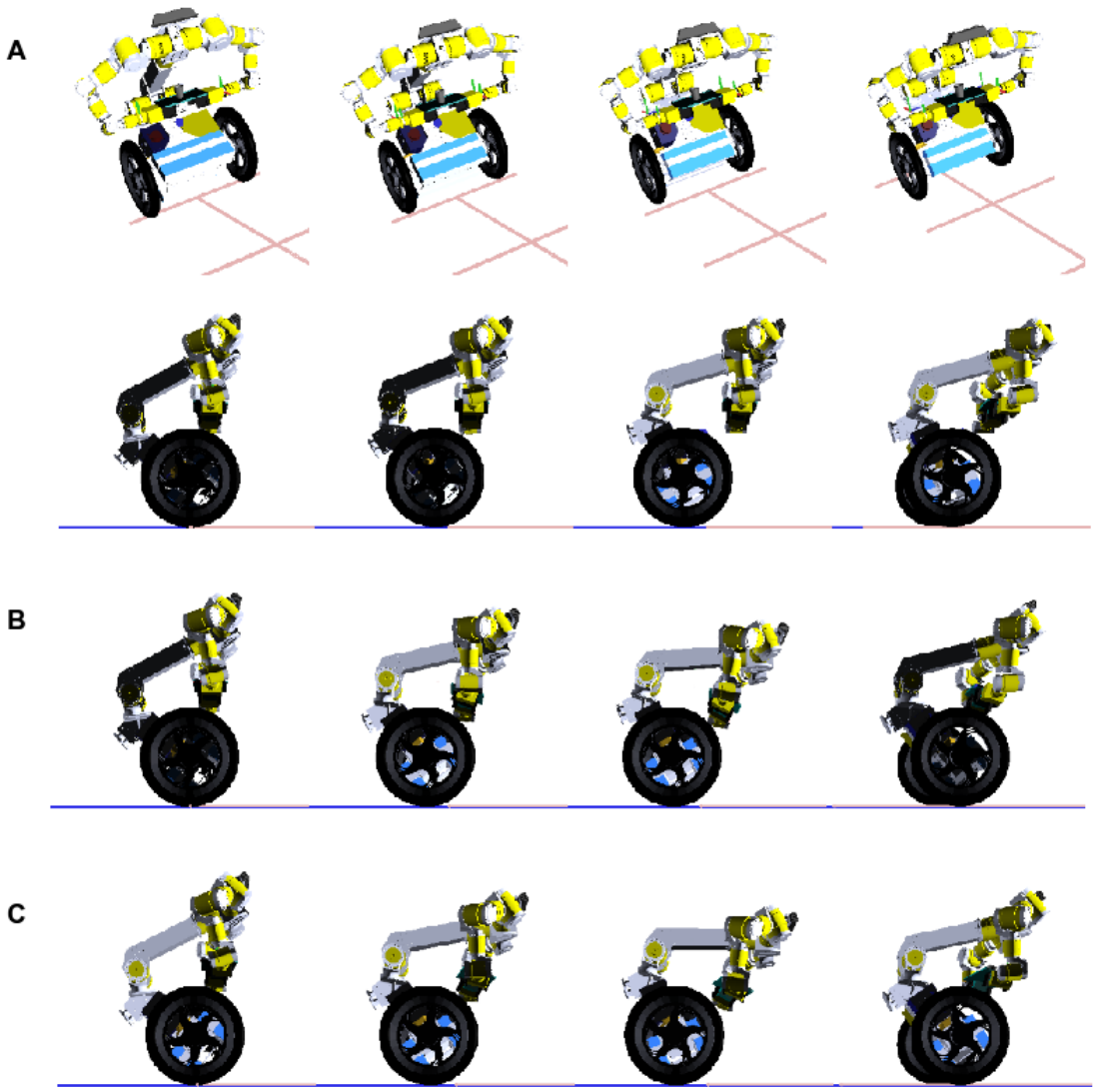



Figure 6.4: Whole-body control of Krang unifying locomotion with end-effector orientation control to carry a tray with a cup. MPC based high-level controller in all cases. A) ID based low-level controller B) Same as A with locked waist joint. C) Low-level controller with IK based arm control and ID based lower body control (locked waist joint).

putation outperformed purely parametric based rigid-body dynamics torque computations as shown in Figure 6.6. Joint 7 is not included because we are not interested in performing learning on this joint due to its relative ease of control through feedback controller. In the joint-space, the normalized mean-square error is computed and the ratio of improvement is tabulated in Table 6.2.

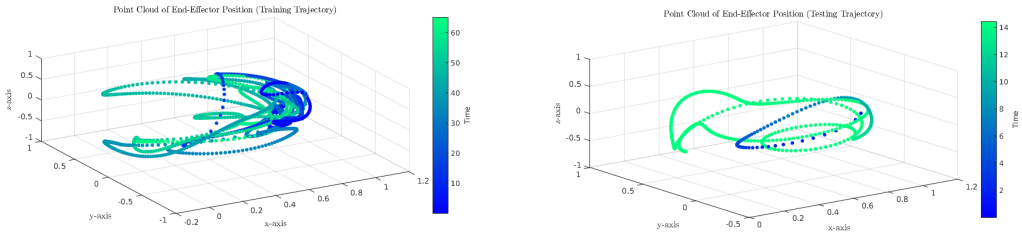



Figure 6.5: Point cloud data of end-effector positions for data collection of training and testing sets.

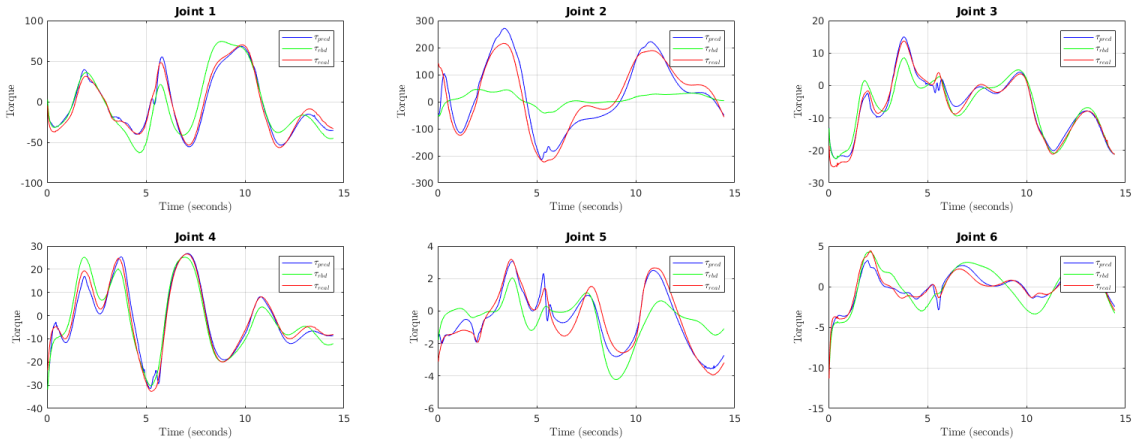



Figure 6.6: Red - measured torque, Blue - GP-predicted torque, Green - RBD-based torque. The GP-predicted torque matches the measured torque fares much better than RBD-based torque.

| Joint 1 | Joint 2 | Joint 3 | Joint 4 | Joint 5 | Joint 6 |
|---|---|---|---|---|---|
| 17.4 | 33.5 | 5.8 | 5.1 | 12.3 | 2.01 |

Table 6.2: Table with normalized mean squared error for each joint. The error is computed between actual torque and predicted torque.

## 6.3 Hardware Demonstration

In this section, we go through details of implementation for hardware experiments. Two stages of work was completed to implement the code successfully for the hardware: Development of the simulation-based verification platform that has the same interface as that provided by the operating system for access to the hardware components, and secondly extension of the legacy application program, that was used for balancing earlier, to incor-

porate the whole-body control algorithm described in the preceding chapters.

### 6.3.1 Simulation-based Verification Platform

Golem Krang is inherently unstable. To stabilize, we used LQR-based controller in our prior work. In order to implement whole-body control framework, we are replacing this working controller with our own high-level controller that is based on MPC. Since the code for MPC is complex to implement, it is absolutely essential that the program is verified thoroughly before it is deployed to replace the basic LQR controller. In order to perform a thorough verficiation, we developed a simulation-based architecture, which is briefly described in this section.

The platform, Golem Krang, is equipped with an on-board computer (Fig 6.7), and precise position encoders for all joints. Real-time sensing and control is possible with many dedicated CAN-based physical interfaces, and the real-time operating system installed on the computer. The driver for hardware interface with each device is installed on the OS with libraries for interfacing with the hardware. Programs running in the background (daemons) are written in-house to interface with the hardware. These daemons themselves allow access via a publish-subscribe architecture. For this purpose the inter-process communication (IPC) suite used in Golem Krang is ACH [71]. An alternative choice is ROS, even though ROS does not currently offer real-time guarantees for inter-process communication and suffers from what is called the head-of-line problem. ACH was designed to avoid these issues.

ACH allows for channels that allow multiple processes to write to and read from. The list of channels are shown in figure 6.8. Each daemon that is dedicated to communicate to a sensing hardware is publishing data to state channels at a specific frequency (currently 30 Hz for all channels). Daemons dedicated to actuators are also listening to command channels at the same frequency. So an application program can be written that controls the hardware by sending data to and receiving data from these ACH channels. This modular-

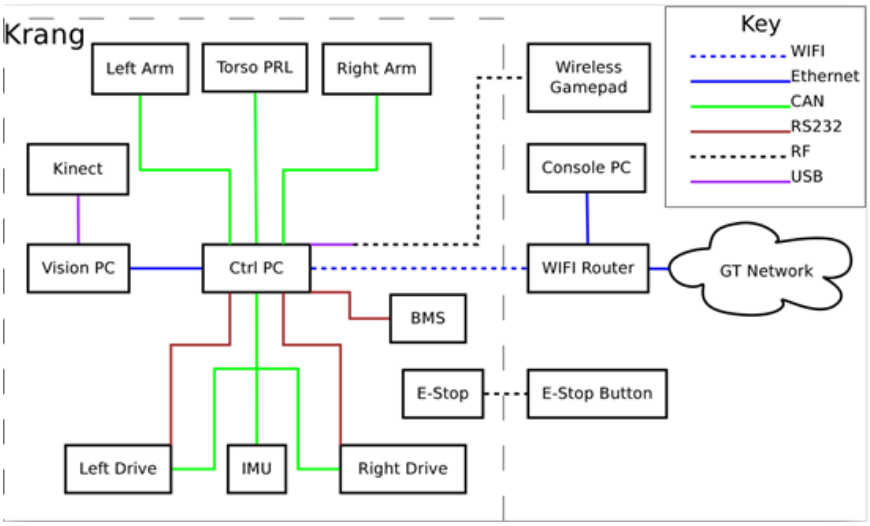



Figure 6.7: Krang electronics

ization is typical in robotic software architectures.

We made use of this modularization to develop our simulation based verification platform. The verification program used DART engine to simulate the dynamics of the physical hardware. And several threads are implemented, each simulating the behavior of the daemons that allow interface with this hardware (Fig 6.8) . Sensing threads will publish state data on their respective state channels in the same format as their corresponding daemons. Similarly, motor control threads read commands from their respective command channels, which are then executed by the physics simulation of the hardware. This platform was used to perform extensive investigation of the design of the application program that implements the whole-body control algorithm.

### 6.3.2 Application Software Design

The application program written for performing whole-body control has to deal with the issue that the robot is initially in the sitting pose when the program starts. In our prior work (figure 6.9), a state machine in implemented which has six different *modes*—a term we use to refer to the states of the state machine to distinguish it from *state* of the dynamics. The modes are as follows:

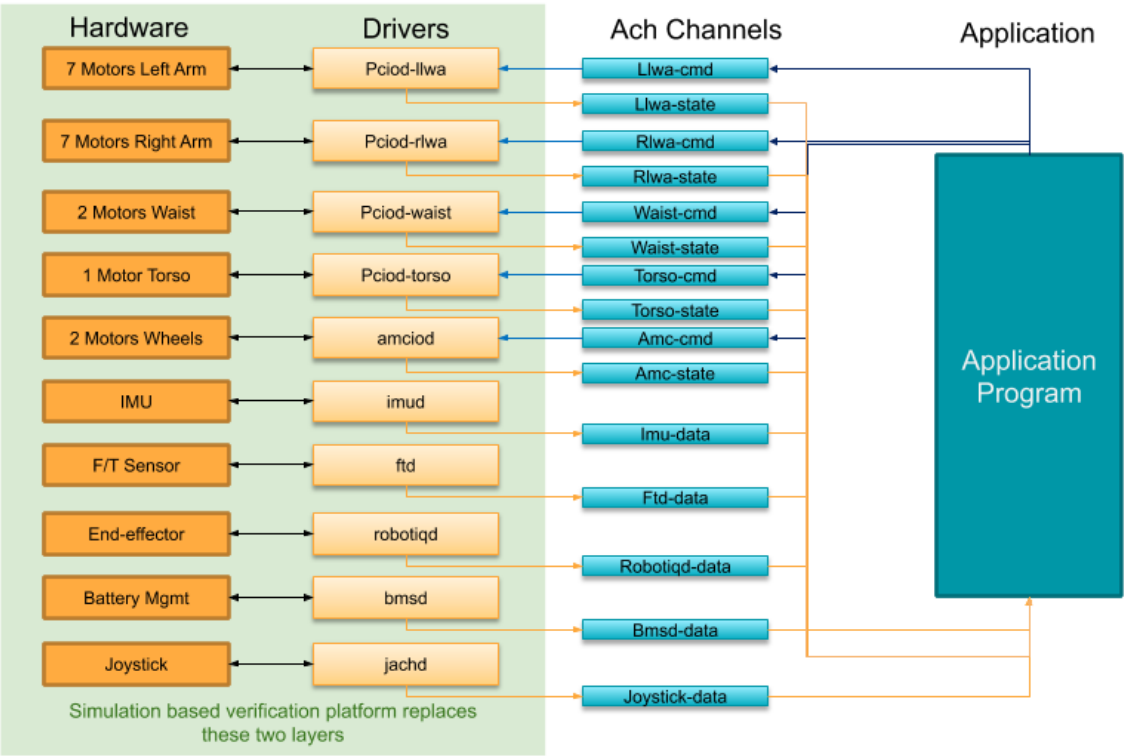



Figure 6.8: Krang software architecture

- Ground Lo: The physical robot is in the sitting pose. Waist angle is small (meaning the angle between spine link and the base link are is smaller than a certain threshold). Controller allows joystick based navigation on ground by controlling heading and spin motion of the wheels.

- Ground Hi: The physical robot is in the sitting pose, with larger waist angle. Same controller as in "Ground Lo", but with different gains to cater for change in ground friction due to less weight of the body carried by the wheels, and more by the caster wheel (which provides the third support in the sitting pose).

- Balance Lo: The physical robot is dynamic balancing itself. Waist angle is small. Controller actively balances the robot while allowing joystick based navigation on ground.

- Balance Hi: The physical robot is dynamic balancing itself. Waist angle is high.

Same controller as in “Balance Lo”, but with a different set of gains to cater for change in dynamics of the robot due to distance of the CoM being larger.

- Stand: The mode in which the controller attempts to stand up. Controller only attempts to come to the balanced position and does not allow joystick based navigation. Once balancing is achieved, it automatically transitions to “Balance Lo” mode.

- Sit: The mode in which the controller attempts to sit down. Controller only attempts to come to the sitting position and does not allow joystick based navigation. Once successfully sit, it automatically transitions to “Ground Lo” mode.

In figure 6.9, the modes and events are drawn. Internal events are labeled with green color, and external events with pink. Internal events are triggered based on internal state of the robot. External events are mapped to joystick and keyboard input so that the user can trigger them.

In order to demonstrate the whole-body control abilities, we make use of this existing state machine to manage standing up from the initial sitting pose. Once stood up and balanced, we introduce another set of modes and events to deal with the requirements of whole body control. This is shown in figure 6.10. Apart from the main thread, we have a DDP thread with its own state machine. The DDP thread is in “Idle” mode in the beginning. If “Start MPC” event is triggered, the main thread remains in its “Balance Lo/Hi” mode, while DDP thread switches to “Compute Traj” mode. In this mode it computes the full horizon trajectory using DDP for the high level controller. Then it switches to “Traj OK?” mode, where it just waits for the user to accept or reject the computed trajectory, that has been displayed on the screen. In the event of “User Demands Recomputation”, DDP thread will start “Compute Traj” all over again. In the event of “User Accepts Trajectory”, the DDP thread switches to “DDP for MPC” mode, where it performs trajectory optimization for a short horizon on a continuous basis for the main thread which has switched to “MPC” mode upon “User Accepts Trajectory”. The main thread obviously performs MPC

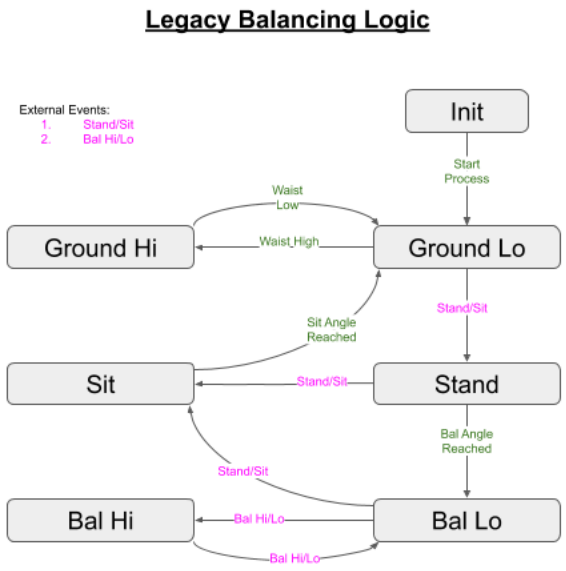



Figure 6.9: State machine in the legacy balancing code

in "MPC" mode. User can "Stop MPC" at any time, in which case main thread reverts back to its previous "Balance Lo/Hi" mode, and DDP thread reverts to "Idle".

In the "MPC", the main thread will control the full body during each sampling period of the receding horizon controller, by calling QP based optimization routines for full body control as described in detail in earlier chapters.

### 6.3.3 Result

To demonstrate a behavior similar to tray-carrying, we have applied this technique on the hardware for the task of carrying a tennis ball on a tray mounted on the end-effector (fig 6.11). The high-level controller in the hardware demo is not MPC, because of limitations in predictive model at a frequency of 30 Hz. Our proposed technique allows the use of any technique in the high-level controller that has the ability to predict the trajectory of the CoM. In order to demonstrate the whole-body behavior, we made use of the existing LQR controller, whose trajectory we learned via data collection through several runs. This enabled us to use the existing LQR controller in the hardware as a high-level controller as we are able to predict the trajectory of the CoM that is needed as reference by the low-level

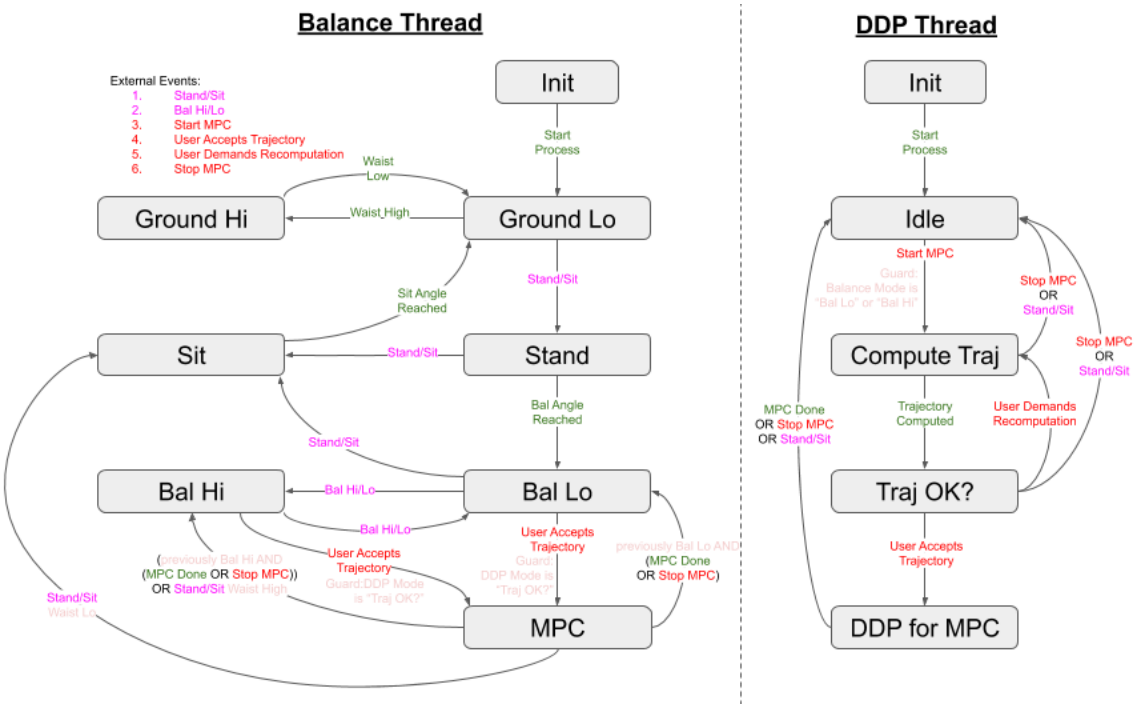



Figure 6.10: State machine for MPC

controller. At the low-level control, we have locked the waist joint, and we use inverse kinematics on the arms with inverse dynamics on the lower body to perform whole-body control in order to perform orientation control of the end-effectors as we follow the CoM trajectory.

To visualize the effectiveness of the technique, we have used a tennis ball on the tray carried by the end-effectors. In the absence of whole-body control, the ball rolls forward and falls down. When whole-body control is enabled, the ball maintains its position on the tray and is prevented from rolling forward and falling down (fig 6.11).

## 6.4 Conclusion

In this chapter, we demonstrated the effectiveness of the presented approach for the basic whole-body control task of maintaining end-effector position and orientation as the robot moves its center of mass to induce motion of the wheels on the ground.

- 7-DOF Planar WIP robot simulation using dynamic models derived using symbolic

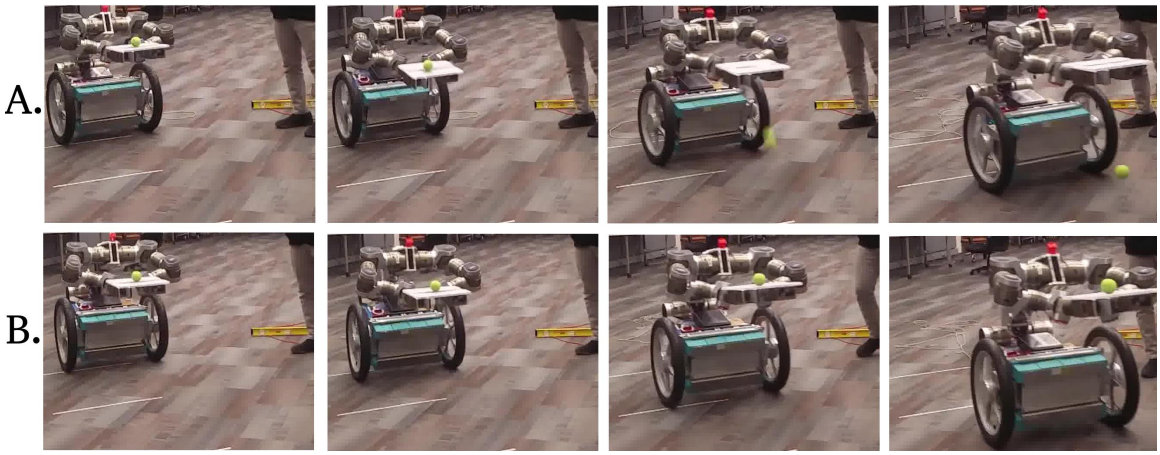



Figure 6.11: Golem Krang carrying a tennis ball on a tray A. In the absence of whole-body control, the ball ends up rolling forward and falling down B. When whole-body control is activated, body joints participate in maintaining end-effector orientation as the robot moves forward by tilting its center of mass

toolbox from MATLAB

- 19-DOF full robot simulation using off-the-shelf physics simulation toolbox, DART
- The physical hardware of the robot Golem Krang

For implementation on the hardware, we discussed the details of the simulation-based verification platform that is ued to verify the correctness of risky stabilization and control techniques. We also discussed how the whole-body control framework is designed within the context of a broader state machine that allows standing, sitting and navigation in other modes of control, and performs whole-body control in the newly introduced mode for the purpose of our experiments.

# CHAPTER 7

# CENTER OF MASS ESTIMATION OF WIP HUMANOIDS

All formulations presented so far all rely on the assumption of perfect knowledge of the center of mass. Even if we use inverse kinematics to control the arms of the robot, thus avoiding the need to know all dynamic parameters of the arms perfectly, we still need to know their CoM parameters, *i.e.*, masses and local center of masses, to find the center of mass of the full robot accurately. This is because trajectory planning and control in the high-level controller, as well as the CoM task execution in the low-level controller, both rely on the assumption of perfect knowledge of the current position of the CoM of the full robot. This makes the CoM parameters especially important compared to the rest of the model parameters such as inertia tensors and frictions. In this chapter, we discuss a novel approach to find the CoM parameters of the robot through efficient data collection and learning. We also propose the use of an online balancing technique, Active Disturbance Rejection Control (ADRC), in order to perform this data collection online, by balancing the robot despite a bad initial estimate of the CoM. But, we have observed the ability of this technique to be limited in practice to achieve this objective. Hence, in the final pipeline, we have collected the data using offline methods, and used our learning technique on this data to converge to a good CoM estimate.

## 7.1 Introduction

Combining the maneuverability of a two-wheeled mobile platform and the dexterity of robotic arms, humanoid Wheeled Inverted Pendulum (WIP) robots present novel challenges to the robotics research community. Humanoid robot stabilization is fundamental to keep the robot safe and for the robot to accomplish higher-level objectives. Furthermore, keeping a WIP, such as the one presented in Fig. 7.1, balanced is a fundamental task in which

the controller needs to be constantly working and thus should be energy efficient [73]. Stabilization is usually accomplished through the control of a simplified two Degree of Freedom (DoF) model which summarizes the Center of Mass (CoM) of all the joints into one as shown in Fig 7.2. This simplification is usually done for both WIP humanoid robots [74, 75, 62], as well as for legged humanoids [76, 77, 78]. All frameworks presented to stabilize WIP robots consider that the mass and CoM for each of the joints is accurately known [79, 80, 81] to compute the simplified two DoF WIP model. However, the mass and real location of the CoM is difficult to obtain, as robot systems can be complex and they might change throughout time. The discrepancy in the parameters of the robot affects the controller's performance, diminishing the robot's dexterity and increasing the power consumption.

Regarding these uncertainties in the model, one common control methodology uses the Modern Control Paradigm [82] which focuses on the modeling of the system as, $\ddot{y} = f(y, \dot{y}, w, t) + bu$, where $y$ is the position output and $w$ is an unknown input force. Once the system is modeled, it is approximated to a linear, time-invariant and disturbance-free model, to design a control law. This approach relies on the model approximation $\bar{f}(\dot{y}, y)$ to be "close enough" to the real model in the neighborhood of the operation point. In [82] and later in [75], Extended State Observers (ESOs) are used to estimate the modeled uncertainties and improve the control of the systems. The approach used collapses all the uncertainties and external forces under one element which is later eliminated through feedback control. From an online learning approach, commonly used models rely on the knowledge and accuracy of the CoM [83, 84, 85]. Very few have worked on model parameter estimation such as [86, 87], but focus more on the estimation of external parameters such as terrain coefficient or external forces than on the robot itself. Finally, recent research involving mobile manipulators has focused on the use of Active Disturbance Rejection Control (ADRC) [88, 89, 90] to control systems which use external uncertainties to conduct feedback control.

Figure 7.1: Full Body of our WIP Humanoid.

Our approach improves our model parameter estimation using the knowledge of the ESO through online learning. The goal of this framework is to create models that are improved upon by real-world systems and data. Given a model of our system, we want to improve the values of the parameters by measuring the disturbances of the system when it is not subject to external forces. Then, as the robot changes its joints position, we are able to update our parameters in an online fashion. To accomplish this task, we propose the following methodology. Given an initial estimation of the parameters of our model $\beta_0$, we use ADRC [82] to estimate the error between the parameters estimated CoM and the real one for different joint configurations. This error is used to update our knowledge of the model parameters through gradient descent. We show that this methodology works, but it might take numerous positions to converge. Thus, we propose a meta-learning algorithm to find the poses which induce the largest gradient step for gradient descent. The main contributions of this works are: *i)* a novel use of the ESO and ADRC to estimate the error in the model parameters; *ii)* an online learning algorithm to update and improve our model parameters; *iii)* a meta-learning framework to improve the speed and accuracy of our learning algorithm; *iv)* and preliminary results on a real robot with 19 DoF that show the improvement of the system's performance.

This paper is organized as follows. Section 7.2 presents the WIP robot and the methodology, as well as discusses the learning, meta-learning, and ADRC techniques. Sections 7.3 and 7.4 describe and present the different simulations and experiments. Finally, section 7.5 presents the conclusions of our work.

## 7.2 Methodology

The goal of the proposed approach is to improve the CoM estimate of a WIP Humanoid. A WIP Humanoid is a highly redundant manipulator mounted on a differential wheeled drive able to dynamically balance itself in an inverted pendulum configuration (Fig 7.2). A good estimate of the CoM is important for any approach to control dynamically balancing humanoids. This is because the balancing task requires the CoM's ground projection to always lie in the support polygon. The support polygon of a WIP is a rectangle on width equal to the distance between the wheels and a small length given by the compression of the wheels against the ground. This support polygon is very thin, hence is important to decreasing the room for errors in CoM estimates compared to, say, bipedal humanoids where support polygons are much larger.

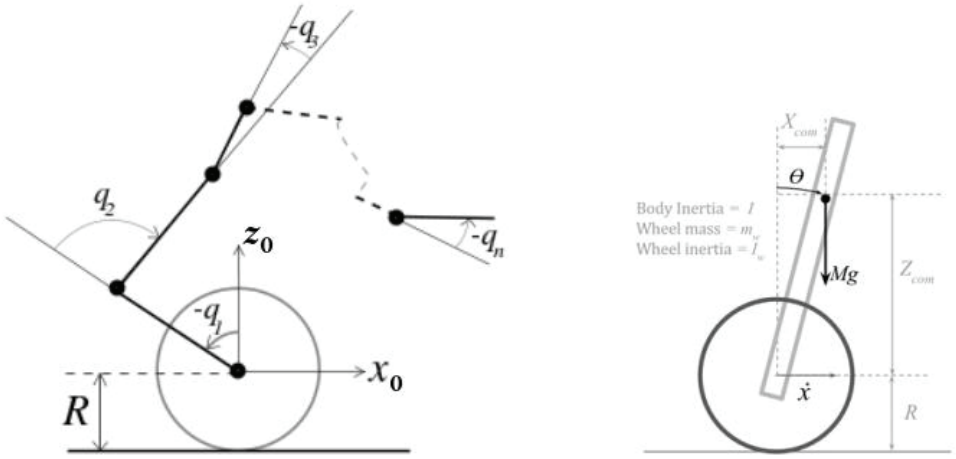
Let us define frame $0$ as the frame where the origin is located at the midpoint between the wheels with its $x$-axis always along the heading direction and $z$-axis always vertical. We are interested in the coordinates of the CoM of the body in this frame. Specifically, we want the $x$-coordinate of body CoM in this frame to be zero in order to balance the robot. Homogeneous coordinates of body CoM in frame $0$ are given by

$$X_{com}(q) = \begin{bmatrix} x_{com}(q) \\ y_{com}(q) \\ z_{com}(q) \\ 1 \end{bmatrix} = \frac{\sum_i^L m_i X_i^0(q)}{\sum_i^L m_i} = \frac{\sum_i^L m_i T_i^0(q) X_i^i}{\sum_i^L m_i}$$

where we are interested in the $x$ component of the CoM

$$x_{com}(q) = \phi(q)^\top \beta \tag{7.1}$$

and the variables are described in Table 7.1.




(a) (b)

Figure 7.2: Full Body of a typical WIP Humanoid with $n$ links (a) 2D Simplified Model (b)

$\beta$ is the set of unknown parameters comprising mass and mass times CoM of individual links in the body. This choice of parameters is such that the parameters appear linearly in the model. Treating CoM values independently from masses would make the model quadratic in parameters. Improving the estimate of the body's CoM entails improving our

Table 7.1: System variables.

| Variable | Description |
|---|---|
| $L$ | number of links in the body |
| $q$ | $\begin{bmatrix} q_1 & ... & q_L \end{bmatrix}^\top$ position of all joints in the body |
| $m_i$ | mass of link $i$ |
| $X_i^0(q)$ | is CoM of link $i$ expressed in frame 0 |
| $X_i^i$ | $\begin{bmatrix} x_i & y_i & z_i & 1 \end{bmatrix}^\top$ local CoM of local frame $i$ |
| $T_i^0(q)$ | transformation from frame $i$ to frame 0 |
| $\beta$ | $\begin{bmatrix} m_1 X_1^{1\top} & ... & m_L X_L^{L\top} \end{bmatrix}^\top \in \mathcal{R}^{4L}$ |
| $\phi(q)$ | $\begin{bmatrix} \phi_1(q) & ... & \phi_{4L}(q) \end{bmatrix}^\top$ feature vector of known geometric functions of $q$ |

knowledge of $\beta$. One way to achieve this is to disassemble the robot into individual links and perform physical measurements for mass and CoM of each link. This is tedious and hence undesirable. However, the fact that the CoM model is linear in $\beta$ allows us to use linear regression or gradient descent to improve our model parameter estimates. We choose gradient descent because of its ability to enforce physical consistency constraints. For one, it converges to $\beta$ values in the neighborhood of the initial $\beta$ which are more likely to be physically consistent as opposed to linear regression which might learn solutions that fit the data but are physically nonsensical. Secondly, constraints such as total body mass can be explicitly enforced in the learning process through the use of Lagrange multipliers. The details of this appear in Section 7.2.1.

These learning techniques rely on our ability to collect data for poses $q$ and corresponding values of outputs $x_{com}(q)$. The simplest way to collect this data is to make use of the fact that in the ideal case, $x_{com}(q) = 0$ when the robot is in a balanced state. Assuming that all joints in the body shown in Fig. 1b can be locked at a specific pose $\{ q_2, ..., q_L \}$, there exists a position for $q_1$ (the base link) that can balance the robot. We can collect data offline by manually moving $q_1$ such that the robot is in a balanced state. However, this is again tedious; performing the same job online would avoid this labor. To this end, we propose ADRC [75] to balance the robot despite a bad estimate of body CoM, the details of which appear in Section 7.2.3. One may ask: Why the need to improve the CoM model if there already exists a controller that is able to stabilize the robot despite a bad CoM estimate? The answer to this is twofold: Firstly, ADRC achieves balancing but is inefficient, i.e. it takes more time and aggressive control inputs to stabilize a bad estimate of CoM. Secondly, ADRC works only when controlling a single rigid link on wheels which is the case when body joints are locked. If however the joints are unlocked, more complex controllers are needed that rely on an accurate estimate of the CoM.

We have so far discussed how to obtain the value of $x_{com}(q)$ at any give pose $q$. It is important to determine what poses at which we should collect this data. This is because

with a highly redundant system, the configuration space is too large and relying on arbitrary poses may make the learning process inefficient and time-consuming. We choose poses such that every next pose causes the largest average gradient descent step over a large set of randomly chosen erroneous $\beta$ estimates. This is discussed in Section 7.2.1.

### 7.2.1 Learning Algorithm

For the learning algorithm, we make use of gradient descent. The objective function to be minimized is determined based on the fact the $x$-component of CoM should be zero in a balanced pose. In order to make the cost function locally convex with respect to $\beta$, we aim to minimize the square of the $x$-component of CoM.

$$\begin{aligned} J(\beta) &= \tfrac{1}{2}\left[x_{com}(q;\beta)\right]^2 \\ &= \tfrac{1}{2}\beta^\top \phi(q)\phi(q)^\top \beta \end{aligned} \tag{7.2}$$

where we have made use of the definition of $x_{com}$ in (7.1).

The gradient with respect to $\beta$ will therefore be

$$\nabla_\beta J(\beta) = \phi(q)\phi(q)^\top \beta \tag{7.3}$$

The update step will be

$$\beta_{t+1} \leftarrow \beta_t - \eta \nabla_\beta J(\beta_t) \tag{7.4}$$

where $\eta$ is the step-size, which is a hand-tuned parameter. We begin with an initial estimate of $\beta$. As data for the new balanced pose $q$ is collected, we make use of the gradient update step in (7.4) to improve $\beta$ estimates. This is repeated until $\phi(q)^\top \beta$ consistently drops below a threshold $x_{tol}$ for a few iterations.

### 7.2.2 Meta-Learning Algorithm

We also deal with the problem of determining a training set of poses that makes the learning process efficient or less time-consuming. For robots with many Degrees of Freedom, the configuration space is huge and choosing an arbitrary set of training poses will likely make the learning inefficient. We determine this training set offline, only using the model in simulation, using the algorithm presented in Algorithm 2. The algorithm requires a large pool of randomly generated balanced and safe poses $\bar{q} \in \mathcal{R}^{n_{DOF} \times n_{poses}}$. A balanced pose is one where a "real" robot (*i.e.*, with $\beta$ values we pretend to be real) is balanced. A safe pose is one where the robot does not collide with itself or the ground, and the joint values are within their physical limits. We precompute the numerical values of the feature vector $\phi(q)$ evaluated at each pose in $\bar{q}$ and store them in $\Phi \in \mathcal{R}^{dim(\beta) \times n_{poses}}$. The algorithm also requires a set of randomly generated erroneous $\beta$ vectors: $\bar{\beta} \in \mathcal{R}^{dim(\beta) \times n_\beta}$. This is done by choosing values of $\beta$ vectors that cause $x_{com}$ estimate errors in estimating the "real" robot's CoM to be of the same order as is observed in the physical system. The key step in

**Algorithm 2** Pose Filtering

**Input:** Set of randomly generated safe & balanced poses: $\bar{q} \in \mathbb{R}^{n_{DOF} \times n_{poses}}$,
Set of $\phi(q)$ evaluated at each given pose: $\Phi \in \mathbb{R}^{dim(\beta) \times n_{poses}}$,
Set of randomly generated erroneous $\beta$s: $\bar{\beta} \in \mathbb{R}^{dim(\beta) \times n_\beta}$
**Output:** Filtered set of poses: $\widetilde{q}$

1: **repeat**
2: $i^* \leftarrow \underset{i \in \{1,...,n_{poses}\}}{\operatorname{argmax}} \sum_k^{n_\beta} |\Phi_i^\top \beta_k|$
3: $\widetilde{q} \leftarrow [\widetilde{q} \quad \bar{q}_{i^*}]$
4: $\phi^* \leftarrow \Phi_{i^*}$
5: $\beta_k \leftarrow \beta_k - \eta \, \phi^* \phi^{*\top} \beta_k \quad \forall \quad k \in \{1, ..., n_\beta\}$
6: $\Phi \leftarrow \Phi \setminus \Phi_{i^*}$
7: $\bar{q} \leftarrow \bar{q} \setminus \bar{q}_{i^*}$
8: **until** $|\phi^{*\top} \beta_k| < x_{tol} \quad \forall \quad k \in \{1, ..., n_\beta\}$ for last few iterations
9: **return** $\widetilde{q}$

Algorithm 2 is step 2 where the pose that causes the largest average error on all erroneous $\beta$'s is chosen to be added to the filtered set of poses $\widetilde{q}$ which is the output of the algorithm. This pose is also used to perform gradient descent on all $\beta \in \bar{\beta}$ (step 5). We choose the

pose that causes the largest prediction error over the updated set $\bar{\beta}$ in each iteration because it is the most informative for the learning process. The learning process stops when the prediction errors due to all $\beta \in \bar{\beta}$ consistently fall below some tolerance $x_{tol}$ for a set number of iterations.

Even though the set of poses generated from meta-learning were acquired from different $\beta$s than that of the real robot, these poses generated a large error that then helped our entire $\bar{\beta}$ set to converge. If our robot's initial $\beta$ is in or even close to the set $\bar{\beta}$, these poses should have a similar effect and cause it to converge.

### 7.2.3 Online Data Collection

We now discuss the problem of balancing the robot despite a bad estimate of body CoM to obtain data points for the learning process. Given that body joints are locked at the desired pose $\{ q_2, ..., q_L \}$, the robot is equivalent to a single rigid link on two wheels, to be balanced by manipulating the base link $q_1$ and the wheels. We propose the use of ADRC [75] for this purpose. This approach for balancing control of a WIP Humanoid is originally intended to handle disturbances represented by a torque $\tau_D$ about the wheel axis. To see how this approach is applicable for our case, we can imagine a virtual robot that has $\beta$ values equal to our current bad estimate and is experiencing a disturbance torque such that the effective CoM of the virtual system has shifted to the real CoM of the physical system. Thus the problem of controlling a robot with a bad CoM estimate is equivalent to one experiencing a disturbance torque about its wheel axle.

A brief explanation of the technique as it applies to our system is as follows. Linearizing the dynamics of WIP Humanoid with its joints locked at pose $q$ in a 2 DoF system

$$\dot{X} = \frac{d}{dt}\begin{bmatrix} x & \dot{x} & \theta & \dot{\theta} \end{bmatrix}^\top = A(q)X + B(q)\tau_w \tag{7.5}$$

where

$x, \dot{x}$ = position and heading speed of the robot

$\theta, \dot{\theta}$ = ang. position and speed of CoM about wheel axis

$\tau_w$ = sum of torques applied on both wheels

$$B(q) = \begin{bmatrix} 0 & 0 & b_x(q) & b_\theta(q) \end{bmatrix}^\top$$

Note that $A$, $b_x$ and $b_\theta$ are functions of parameters such as CoM distance from wheel axis and body inertia that are dependent on $q$. Applying LQR on this pose-dependent linearized system $(A(q), B(q))$ results in pose-dependent feedback gains

$$F(q) = \begin{bmatrix} F_x(q)^\top & F_\theta(q)^\top \end{bmatrix}^\top = \text{LQR}(A(q), B(q)) \tag{7.6}$$

Treating $\dot{x}$ and $\dot{\theta}$ dynamics as two independent subsystems by following [91], we can find the control inputs as

$$u_x = -F_x(q)^\top \begin{bmatrix} x & \dot{x} \end{bmatrix}^\top \quad u_\theta = -F_\theta(q)^\top \begin{bmatrix} \theta & \dot{\theta} \end{bmatrix}^\top$$

The standard feedback control setting for WIP systems has the control input defined by $\tau_w = u_x + u_\theta$. However, the key to perform active disturbance rejection is to estimate the numerical value of dynamic disturbances in the two subsystems, $\hat{f}_x$ and $\hat{f}_\theta$, due to the inaccurate CoM estimate and compensate for those disturbances using feedback linearization:

$$\tau_w = \left( u_x - \frac{\hat{f}_x}{b_x(q)} \right) + \left( u_\theta - \frac{\hat{f}_\theta}{b_\theta(q)} \right) \tag{7.7}$$

Here, $\hat{f}_x$ and $\hat{f}_\theta$ are estimating the dynamic disturbances $f_x$ and $f_\theta$ in the subsystems appearing in state space representation of the dynamic model

$$\begin{aligned} \ddot{x} &= f_x(X, q, \tau_D, u_\theta) + b_x(q) u_x \\ \ddot{\theta} &= f_\theta(X, q, \tau_D, u_x) + b_\theta(q) u_\theta \end{aligned} \tag{7.8}$$

The estimates are found using Extended State Observers

$$\frac{d}{dt}\begin{bmatrix}\hat{\theta}\\ \hat{\dot{\theta}}\\ \hat{f}_\theta\end{bmatrix}=\begin{bmatrix}\hat{\dot{\theta}}+l_{\theta 1}(\theta-\hat{\theta})\\ \hat{f}_\theta+l_{\theta 2}(\theta-\hat{\theta})+b_\theta u_\theta\\ l_{\theta 3}(\theta-\hat{\theta})\end{bmatrix},\qquad \frac{d}{dt}\begin{bmatrix}\hat{x}\\ \hat{\dot{x}}\\ \hat{f}_x\end{bmatrix}=\begin{bmatrix}\hat{\dot{x}}+l_{x1}(x-\hat{x})\\ \hat{f}_x+l_{x2}(x-\hat{x})+b_x u_x\\ l_{x3}(x-\hat{x})\end{bmatrix} \tag{7.9}$$

where the observer gains $l_x$ and $l_\theta$ are designed using pole placement.

## 7.3 Simulation Results

We started experiments by simulating our pipeline; we first considered a WIP model with 7 DoF in Matlab and next a WIP with 19 DoF in the 3D Dynamic Animation and Robotics Toolkit (DART) [92]. The former served as a more tractable proof of concept that led into the latter, a more faithful representation of the robot that we will be using during the experiments. In both simulations, we provided the class methods for instantiating an $L$-link WIP model, updating their mass parameters $\beta$, approximating their dynamics, applying control, and visualizing the results. Simulation provided us with two key benefits over hardware: (1) it allowed us to rapidly spawn, control, and respawn our robot in a safe, realistic setting; and (2) it allowed us immediate access to parameters that were otherwise "unknowable", or difficult to obtain. For our system specifically, these parameters are the masses and Center of Masses for individual links, which are both numerous and inaccessible to measurements. To evaluate the performance of our algorithms we instantiated two full $L$-link WIP models

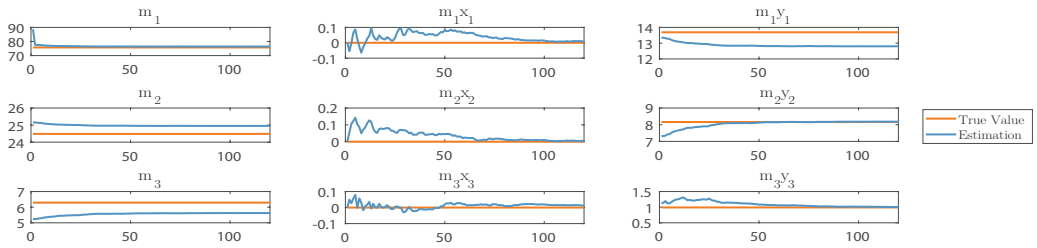



Figure 7.3: Values of the different parameters of the estimated model over 120 different configurations. The red lines show the real value of the parameter for the real robot, while the blue lines show the learned weights.

– a ground truth model and an inaccurate model with an estimation of the parameters of the first robot. These two models served as placeholders for the arm's configuration and mass parameters. We then simplified these two models into their single link representation (Fig. 7.2 - right). In Matlab, using an ODE45 integration loop, we simulated the system dynamics from the ground truth model and then calculated the control signals based on the estimated simplified model. In the DART implementation, the dynamics were updated automatically by the simulator. We first started by tuning our ADRC's LQR gains to be able to control the estimated simplified model to the balance position of the ground truth model. During this process, we iteratively set both models to randomized joint angles on the configuration space. After tuning the controller and observer parameters for each joints configuration, the ADRC would balance the systems to its true balance position, i.e. for a given configuration $q_2, q_3, \ldots, q_L$, the ADRC would find the value of $q_1$ that balanced the system.

### 7.3.1 Gradient Descent Simulation

The offset given by the ADRC for the estimated model was used in a gradient descent algorithm to update our estimated model parameters. Starting with the Matlab simulation, the estimated model was subject to initial noise for the initial estimation of $20\%$ from the real values of the parameters $m_i$, $m_i x_i$ and $m_i y_i$. Since each link had different properties (similar to our experimental robot), the noise perturbation differed; the first link has an approximated mass of $70kg$ which gives a noise around $14kg$, while the third link has a mass of $6kg$ which give us a noise around $1.2kg$. Using **??** we update our $\beta$ for each iteration. A subset of the parameters of $\beta$ are shown in Fig. 7.3.

It can be seen in Fig. 7.3 that our algorithm modifies the $\beta$ vectors, reaching a local minimum. For some parameters (as $m_1$, $m_1 y_2$ or $m_3 y_3$) the estimated values converge to the real values, while for others (as $m_2$, $m_3$ or $m_1 y_1$) the values converge to a constant error. Even though we are finding a local minima and not necessarily the correct values,

we will show that our new estimate of $\beta$ improves upon the initial values. After running different simulations, we notice that while the system always reaches a $x_{CoM}$ error of zero, the weights converge to different values – giving the intuition that the system consists of several local minima.

This method has shown that the approach works in finding a better set of values than the ones we initially started with, but might not get to the global optimum (the real values). We think that this happens because of the nonlinearities of the system and because the $\beta$ vector is not perfectly decoupled to the value of the masses.

### 7.3.2 Meta-learning for Gradient Descent Convergence

As described in section 7.2.2, we simulated 20,000 poses over 500 erroneous $\beta$s, and got a set of 528 poses until the error was $2mm$. Without using the meta-learning algorithm this process takes over 5,000 poses. The result of our simulated learning curve is presented in Fig. 7.4. We tested for several initial erroneous $\beta$s which started with an $x_{CoM}$ error of at least $2cm$ with a standard deviation of $0.5cm$. It can be seen that after $500$ updates using the optimal poses, the mean error decreased to almost $0cm$, specifically the max $\beta$ error decreased to $x_{tol} = 2mm$.

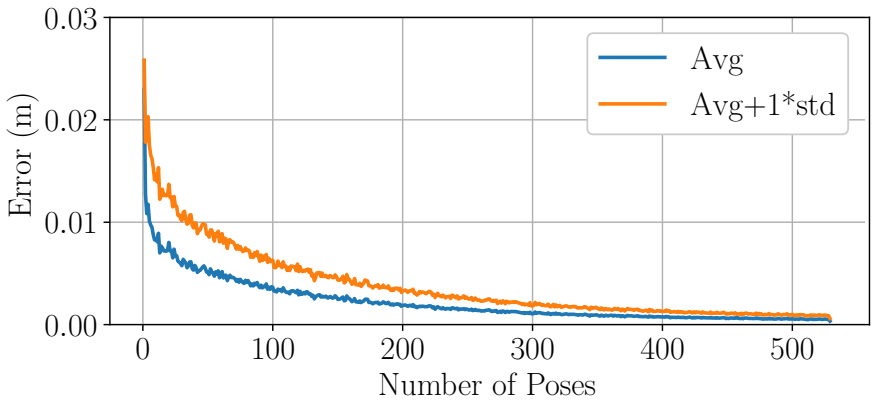



Figure 7.4: Mean Error of several $\beta$s through the learning algorithm for the meta-learned best 500 poses.

## 7.4 Experimental Results

For the robot that we are using, Golem Krang [93], determining its mass model link-by-link is intractable. Furthermore, the summarizing CoM described in 7.2 is difficult to obtain. Instead of extracting the full mass model or CoM estimates, we follow the procedure of other work [94, 95] to evaluate balancing performance. Where the authors analyze more readily observable phenomena, such as distance traveled, time spent stabilizing, and power consumption. In our case, these quantities were used to analyze whether or not subsequent refinements of an initial offset estimation ($\beta_0$) improves the stabilizing control. The physical experiments were separated in two parts: manual data collection and controller efficiency testing. For the first part, we collected data from our subset of pre-determined balanced poses – we manually positioned the robot in the first 236 poses acquired from the meta-learning algorithm in Section 7.2.2 and calculated the error between the real $x_{com}$ and the estimation. We obtained this error by setting the robot to presumed balanced pose (which may not actually be balanced under our inaccurate $\beta_0$), and adjusted the base link angle $q_1$ until the system became balanced. We then separated this data into a training set of 190 poses and a testing set of 46 poses. Then, using the training set, we implemented gradient descent to obtain a series of betas going from $\beta_1, \beta_2, \ldots, \beta_{190}$. For each beta, we computed the errors produced by the remaining balanced poses in the testing dataset; the results are shown in Fig. 7.5. For $\beta_0$, we started with a mean error of $2.5cm$ in the $x_{CoM}$ for the given $46$ poses and a maximum error of $6cm$. With subsequent iterations, the mean error and the maximum error decreased. For $\beta_{190}$ we achieved a mean error of $0.4cm$ with a maximum error of $1.2cm$ for any given pose in the testing set.

For the second part, we used five of our learned $\beta$s to balance the robot in a given pose. Specifically, we looked at the initial balancing action, which involves transitioning between a stable sitting position to an inverted pendulum position. For this action, the robot stands from three points of contact with the ground (two active wheels and a caster

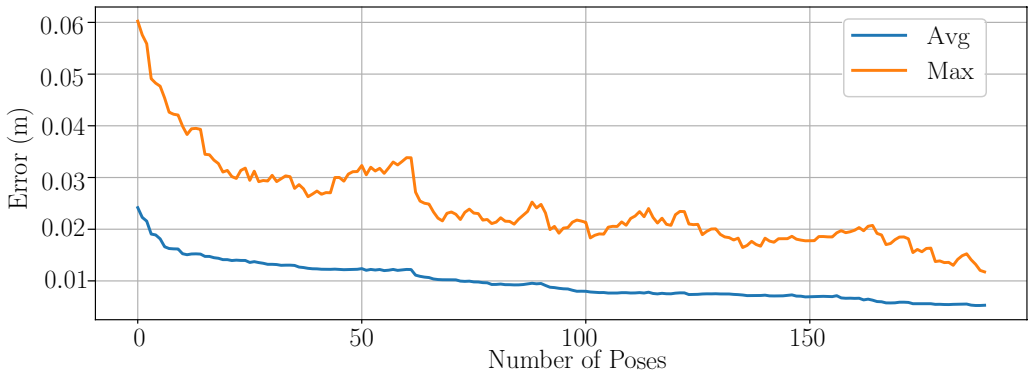



Figure 7.5: Error in the parameters as we update the weights $\beta$ for different random configurations.

wheel). Then it rotates its wheels (at a speed which depends on its CoM estimate) to lift off the caster, and it finally balances as a two-wheeled WIP. The balancing experiments tested different $\beta$ estimates to show how the overall control improves during the transition and steady state of the robot. To investigate the connection between updated $\beta$ vectors and controller performance, we show the results of testing $\beta_{16}$, $\beta_{32}$, $\beta_{64}$, $\beta_{128}$ and $\beta_{190}$. Smaller $\beta$s are not shown, since the robot controller was not able to securely stabilize the system. Additionally, for each $\beta_i$ we tested seven attempts to see the reproducibility of the results.

The instantaneous power consumption of the wheel motors during and after the transition to standing is shown in Fig. 7.6, and a summary of the control performance is presented in Table 7.2. The instantaneous power was calculated by multiplying the torque and angular velocity of the wheels.

As shown in the left column of Fig. 7.6 and in Table 7.2, the peak power consumption decreases with subsequent values of beta. As shown in the right column of the same figure, the number of balancing adjustments (spikes in power consumption) is similarly reduced. For the first $\beta$ values, the system occasionally destabilized and readjusted, whereas the latest $\beta_{190}$ value kept these adjustments and hence overall power consumption to a minimum.

Table 7.2 shows improvement in several quantities that characterize control performance: the initial overshoot position decreases by $44\%$ between the $\beta_{16}$ and $\beta_{190}$ iterations; the resting position decreases by $57\%$; the time until resting decreases by $38\%$; the

Table 7.2: Summary of the control performance under different betas.

| $\beta$ | Max Pos. [m] | Resting Pos. [m] | Time until Resting [s] | Max Power [W] | Avg. Resting Power [mW] |
|---|---|---|---|---|---|
| $\beta_{16}$ | $4.70\pm 0.16$ | $2.49\pm 0.03$ | $11.5 \pm 1.5$ | $145\pm 4$ | $7.83 \pm 1.69$ |
| $\beta_{32}$ | $4.59\pm 0.11$ | $2.67\pm 0.11$ | $10.1 \pm 1.1$ | $133\pm 4$ | $8.85 \pm 7.09$ |
| $\beta_{64}$ | $3.59\pm 0.17$ | $1.53\pm 0.05$ | $7.59 \pm 1.33$ | $63.1\pm 3.3$ | $2.95 \pm 1.04$ |
| $\beta_{128}$ | $2.74\pm 0.07$ | $1.13\pm 0.03$ | $6.80 \pm 1.20$ | $41.3\pm 6.9$ | $2.90 \pm 0.60$ |
| $\beta_{190}$ | $\mathbf{2.61\pm 0.08}$ | $\mathbf{1.08\pm 0.03}$ | $\mathbf{7.09 \pm 1.97}$ | $\mathbf{34.5\pm 13.2}$ | $\mathbf{1.54 \pm 0.25}$ |

peak instantaneous power decreases by $76\%$; and the average power during steady state balancing decreases by $80\%$. Each of our performance metrics improves with more refined mass model parameters. Together, these trends support the claim that the CoM estimation procedure does improve balancing for a WIP.

## 7.5 Conclusion

We have shown that the proposed methodology improves the CoM estimate of a WIP Humanoid and that these improvements translate to improved controller performance. In simulation, using active disturbance rejection control, our robot successfully balances with an inaccurate prior mass model, collects new pose data at balanced positions, and learns from these poses to produce a more accurate CoM estimate. In hardware, we demonstrate that these refined estimates directly translate into improved controller performance. Together, our simulation and hardware results support the claim that our algorithm – a semi-automated, tractable procedure that refines the latent space mass model of a high dimensional system with few physically observable parameters – does improve overall balance. The algorithm was probed in simulation and verified physically on a 19 DoF WIP

robot. Our future work will implement the fully automated estimation pipeline–active disturbance rejection control, balanced pose data collection, and online learning–in an entirely online fashion on the physical robot, where it will improve its parameter estimates through meta-learned poses.

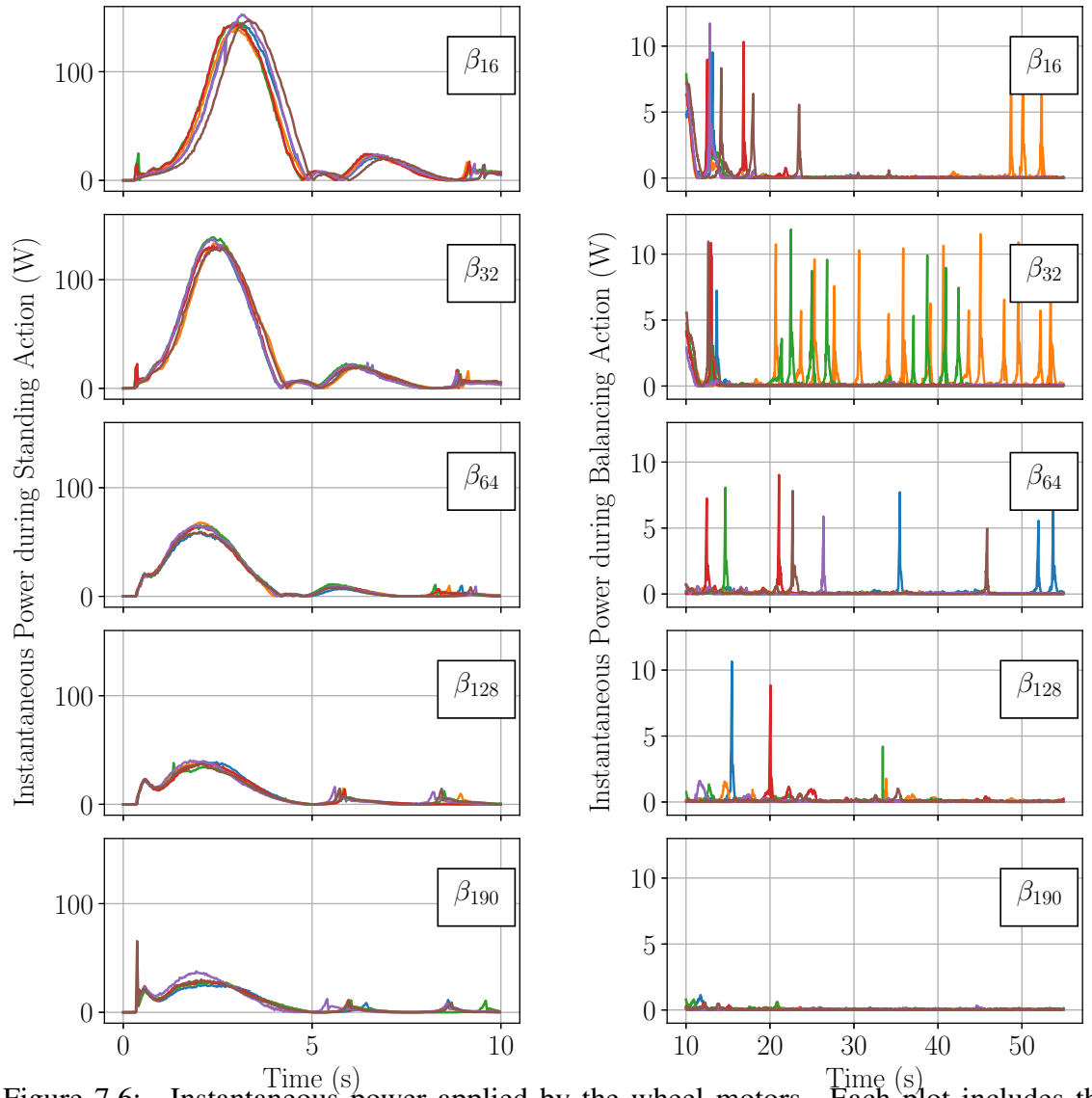



Figure 7.6: Instantaneous power applied by the wheel motors. Each plot includes the results corresponding to 7 independent runs for different values of $\beta$ ($\beta_{16}$, $\beta_{32}$, $\beta_{64}$, $\beta_{128}$ and $\beta_{190}$). The left column summarizes the sitting-standing transition (the first 10 seconds of the experiment), and the right column summarizes the WIP balancing (the subsequent 10 to 60 seconds).

# CHAPTER 8
# FUTURE WORK

It is true that for systems with complex and unstable dynamics, such as humanoids, the use of model-based control within a hierarchical framework remains the tool of choice. This is due to the challenges associated with applying model-free reinforcement learning on such problems, such as sample inefficiency and limits on exploration of state space in the absence of safety/stability guarantees. However, relying purely on physics-based models comes with its own set of problems. For instance, the necessary limits on expressiveness imposed by committing to fixed basis functions, and consequently, their limited ability to learn from data gathered on-line. The hierarchical approach presented in this work is also model-based. These models, although of immense use, are based on assumptions that may not always hold (like rigid bodies, simplified friction behaviors, and linearity and invariability of identifiable parameters). In Golem Krang, all joints are actuated by harmonic drives with high frictions, and the relationship between current commands and actual torques is not reported to be linear. Thus, it violates key assumptions at play in the hierarchical whole-body control framework discussed above.

This gap between theoretical models and real-world dynamics gives rise to a need to incorporate a learning component at some level within the model-based control framework. To address this, the use of semi-parametric representations of our dynamics for predictive control in both levels of the hierarchy is a promising area of investigation. This is because they allow the physics-based models to be incorporated in a data-driven learning framework, which has also been demonstrated to be more sample-efficient compared to purely model-free techniques. Some of the recently proposed techniques include:

1. Semi-parametric Gaussian Process-based approaches to computed-torque control of serial robots [96]

2. Probabilistic Differential Dynamic Programming framework for trajectory planning by high-level controllers [97, 98]

3. Barrier Certificate based safe-learning approaches for data collection to learn the dynamics of inherently unstable systems [99].

The use of these learning components at various stages of control design and hierarchy can potentially deal with model uncertainty in hierarchical control framework presented in this work. The aim is to allow new observations online to improve the control performance. The aforementioned semiparametric approaches make use of Gaussian Processes to represent unknown nonlinearities to be learned while preserving the parametric part. Within our framework, these GP-based approaches can be applied at three stages of the design: Safe-learning for data collection on unstable dynamics, inverse dynamics (ID) using semi-parametric model for torque calculations for low-level control, and trajectory planning and optimization in the high-level controller based on uncertainty in dynamics modeled by GPs.

To this end, the dynamics can be represented using a semi-parametric framework similar to [96] for the low-level controller. This framework relies on the ability of Gaussian Processes to incorporate physics-based non-linear parametric basis functions as priors, while augmenting rich classes of non-parametric kernel functions to learn unknown non-linearities not captured by their parametric counterparts. For learning to overcome model uncertainty in the high-level controller, a semi-parametric approach to trajectory optimization needs to be incorporated that allows prediction of states over a horizon. This is done while taking into account the uncertainty in system dynamics, and simultaneously making use of new data to improve upon the model of the dynamics. To this end, a recent framework proposed by [98] has outperformed the state-of-the-art on sample-efficiency in reinforcement learning demonstrated over toy problems such as simulated quadrotor control and pendulum swingup. The framework uses again a Gaussian Process-based semi-parametric representation of dynamics, and makes use of the uncertainty metric provided by GPs to propagate the states for trajectory rollout. These trajectories are then iteratively

optimized using Differential Dynamic Programming that offers real-time convergence to local optima due to second-order approximation of dynamics, and thus offers real-time control (in a receding horizon framework). To deal with potential computational burden of GPs in both levels of the hierarchy, a recent approach by [98] can be utilized. Finally, to address the problem of collecting data on our inherently unsafe and unstable dynamics, the safe-learning framework of [99] can be used. This framework relies on barrier certificates in the presence of GP-based uncertainties to reason about the safety of exploration in unknown regions of the state space for gathering data for learning.

Some of the key experimental insights from this work and the future directions discussed above are:

1. The hierarchical whole-body control approach performs significantly better than the traditional implementation for WIP humanoid control. For example, in our previous implementation [75], the arms are being manipulated using joint-level PID controllers on trajectories produced by IK. A separate process running an LQR-based balancing algorithm balances the robot by rapidly updating the CoM as the robot changes its pose. In our experience with this implementation, the behavior of the robot is not suited to provide guarantees to higher-level AI algorithms. As the controllers are not unified/coordinated, it is impossible to predict the effects of one on the other, and thus makes it very difficult to plan trajectories for performing complex tasks. As an example, when moving forward, change in pitch of the robot forces the end-effector's position to change if the upper body control is not coordinated with CoM motion. Similarly, during large-force interactions, planning body poses that prevent large shifts in equilibrium positions is only possible with a unified approach. In our simulation-based experiments, we have verified that the hierarchical controller performs better in executing coordinated manipulation and locomotion task.

2. For high-level controllers, the simplification of the robot's body to a single rigid link with an equivalent CoM is sufficient. In the bipedal robots, this simplification is quite

common, but this is the first time a similar simplification is being done for whole-body control of a WIP humanoid. Given that WIP dynamics are more unstable, it was important to establish that a similar simplified model succeeds to capture the essential parts of the dynamics.

Furthermore, based on the proposed learning methods in the preceding paragraphs, we can inquire the following set of questions:

1. Does the safe learning approach proposed by [99] for collecting data on unstable systems and demonstrated on a simulated quadrotor example, generalize to other unstable systems like WIP humanoids, and how well does it work in practice? Are they effective in improving the control policy from the policy at work while collecting data?

2. Can fast approximate GP prediction [98] and QP-based ID on semi-parametric models of a 25 DOF system be made fast enough to allow real-time control of the system?

3. Does performing DDP in the belief space [97] significantly outperform the deterministic implementation of DDP for the high-level controller, in terms of robustness and stability?

Finally, adaptive interaction of controllers operating at different time-scales is a promising area of investigation. The data collected on manual fine-tuning of the performance based on adjustment of various tuneable parameters can help develop a framework for adaptive online change of the weights/gains of the controllers according to requirements of the tasks being performed. For example, when taking a sharp turn, we may like to enforce stiffness in high-level controller objectives to make the low-level controller strictly follow its targets, but we may want to relax this stiffness when, say, moving in a straight line.

## 8.1 Concluding Remarks

In summary, we made the following contributions in this work:

- Deriving the full 3D dynamic model of a typical WIP Humanoid
- Developing a hierarchical control framework in which different tasks can be performed on the WIP platform in a unified fashion potentially resulting in better overall performance
- Showing how quadratric programming (QP) based control techniques [2] can be applied as a low-level controller to isolated manipulator dynamics obtained by elimination of wheel dynamics from the dynamic model
- Analysing the zero dynamics of the system to motivate the use of a simplified Wheeled Inverted Pendulum Model (WIPM) that approximates the inertial properties of the full robot for predictive control by a high-level controller
- Using differential dynamic programming (DDP) and model predictive control (MPC) for generating and controlling CoM trajectory of the WIP Humanoid using predictions of the WIPM. This serves as the high-level controller for providing feasible CoM targets to the low-level controller to ensure effective balancing and locomotion along with the performance of other tasks
- Developing a simulation-based verification platform to test risky new control algorithms before deployment on the hardware
- Implementation of the full pipeline of the whole-body control architecture within an existing framework that supports basic tasks of sitting down and standing up and tele-operated motion-control
- A novel data-driven method to improve estimates of the CoM model of the robot is

presented which uses statistical techniques to determine a good set of sample poses that ensure fast convergence of model parameters using gradient descent

- As future work, we presented concrete set of ideas that can be incorporated within the presented framework to deal with model uncertainty, by making use of semi-parametric representations of dynamics at all levels of the hierarchy. This can enable us to they use physics-based models as priors while retaining the ability to:
  - learn non-linear dynamic effects that were not captured by the model using data gathered online.
  - plan trajectories for the high-level controller in the belief space of the simplified model
  - perform inverse dynamics in the low-level controller using the learned representations of the full model

## REFERENCES


[1] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *International Journal of Humanoid Robotics*, vol. 2, no. 04, pp. 505–518, 2005.

[2] S. Feng, "Online hierarchical optimization for humanoid control," 2016.

[3] K. D. (2001). The segway personal transporter (pt), the first self-balancing, zero emissions personal transportation vehicle., (visited on 04/27/2017).

[4] J. Solis, R. Nakadate, Y. Yoshimura, Y. Hama, and A. Takanishi, "Development of the two-wheeled inverted pendulum type mobile robot wv-2r for educational purposes," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, IEEE, 2009, pp. 2347–2352.

[5] C.-C. Tsai, H.-C. Huang, and S.-C. Lin, "Adaptive neural network control of a self-balancing two-wheeled scooter," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 4, pp. 1420–1428, 2010.

[6] Z. Li and C. Yang, "Neural-adaptive output feedback control of a class of transportation vehicles based on wheeled inverted pendulum models," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 6, pp. 1583–1591, 2012.

[7] H. Hata and T. Takimoto, "Development of the portable two-wheeled inverted pendulum type personal vehicle," in *Control, Automation and Systems (ICCAS), 2014 14th International Conference on*, IEEE, 2014, pp. 1610–1613.

[8] C.-N. Huang, "The development of self-balancing controller for one-wheeled vehicles," *Engineering*, vol. 2, no. 04, p. 212, 2010.

[9] P. Petrov and M. Parent, "Dynamic modeling and adaptive motion control of a two-wheeled self-balancing vehicle for personal transport," in *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, IEEE, 2010, pp. 1013–1018.

[10] L. Vermeiren, A. Dequidt, T. M. Guerra, H. Rago-Tirmant, and M. Parent, "Modeling, control and experimental verification on a two-wheeled vehicle with free inclination: An urban transportation system," *Control Engineering Practice*, vol. 19, no. 7, pp. 744–756, 2011.

[11] Y. Takahashi, S. Ogawa, and S. Machida, "Front wheel raising and inverse pendulum control of power assist wheel chair robot," in *Industrial Electronics Society, 1999. IECON'99 Proceedings. The 25th Annual Conference of the IEEE*, IEEE, vol. 2, 1999, pp. 668–673.

[12] Y. Takahashi, T. Takagaki, J. Kishi, and Y. Ishii, "Back and forward moving scheme of front wheel raising for inverse pendulum control wheel chair robot," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, IEEE, vol. 4, 2001, pp. 3189–3194.

[13] Y. Takahashi, N. Ishikawa, and T. Hagiwara, "Soft raising and lowering of front wheels for inverse pendulum control wheel chair robot," in *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, IEEE, vol. 4, 2003, pp. 3618–3623.

[14] Y. Takahashi and M. Kohda, "Human riding experiments on soft front wheel raising of robotic wheelchair with inverse pendulum control," in *Industrial Technology, 2005. ICIT 2005. IEEE International Conference on*, IEEE, 2005, pp. 266–271.

[15] S. Jeong and T. Takahashi, "Wheeled inverted pendulum type assistant robot: Inverted mobile, standing, and sitting motions," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, IEEE, 2007, pp. 1932–1937.

[16] K. Sasaki and T. Murakami, "Pushing operation by two-wheel inverted mobile manipulator," in *2008 10th IEEE International Workshop on Advanced Motion Control*, IEEE, 2008, pp. 33–37.

[17] T. Murakami and K. Sasaki, "A motion control for two-wheel inverted pendulum type of mobile manipulator," *IEEJ Transactions on Electrical and Electronic Engineering*, vol. 4, no. 2, pp. 192–198, 2009.

[18] C. Acar and T. Murakami, "Multi-task control for dynamically balanced two-wheeled mobile manipulator through task-priority," in *2011 IEEE International Symposium on Industrial Electronics*, IEEE, 2011, pp. 2195–2200.

[19] S. R. Kuindersma, E. Hannigan, D. Ruiken, and R. A. Grupen, "Dexterous mobility with the ubot-5 mobile manipulator," in *Advanced Robotics, 2009. ICAR 2009. International Conference on*, IEEE, 2009, pp. 1–7.

[20] M. Stilman, J. Olson, and W. Gloss, "Golem krang: Dynamically stable humanoid robot for mobile manipulation," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, IEEE, 2010, pp. 3304–3309.

[21] T. Feng, T. Liu, X. Wang, Z. Xu, M. Zhang, and S.-c. Han, "Modeling and implementation of two-wheel self-balancing robot equipped with supporting arms," in *Industrial Electronics and Applications (ICIEA), 2011 6th IEEE Conference on*, IEEE, 2011, pp. 713–718.

[22] H. Fukushima, K. Muro, and F. Matsuno, "Sliding-mode control for transformation to an inverted pendulum mode of a mobile robot with wheel-arms," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 7, pp. 4257–4266, 2015.

[23] B. Dynamics. (2017). Boston dynamics officially unveils its wheel-leg robot: "best of both worlds", (visited on 04/27/2017).

[24] M. Stilman, M. Zafar, C. Erdogan, P. Hou, S. Reynolds-Haertle, and G. Tracy, "Robots using environment objects as tools the macgyverparadigm for mobile manipulation," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, IEEE, 2014, pp. 2568–2568.

[25] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *Robotics and Automation, IEEE Journal of*, vol. 3, no. 1, pp. 43–53, 1987.

[26] L. Sentis and O. Khatib, "A whole-body control framework for humanoids operating in human environments," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, IEEE, 2006, pp. 2641–2648.

[27] L. Sentis, J. Park, and O. Khatib, "Compliant control of multicontact and center-of-mass behaviors in humanoid robots," *IEEE Transactions on robotics*, vol. 26, no. 3, pp. 483–501, 2010.

[28] M. De Lasa, I. Mordatch, and A. Hertzmann, "Feature-based locomotion controllers," in *ACM Transactions on Graphics (TOG)*, ACM, vol. 29, 2010, p. 131.

[29] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *The International Journal of Robotics Research*, vol. 33, no. 7, pp. 1006–1028, 2014.

[30] A. Herzog, L. Righetti, F. Grimminger, P. Pastor, and S. Schaal, "Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, IEEE, 2014, pp. 981–988.

[31] M. Hutter, M. A. Hoepflinger, C. Gehring, M. Bloesch, C. D. Remy, and R. Siegwart, "Hybrid operational space control for compliant legged systems," *Robotics*, p. 129, 2013.

[32] M. Hutter, H. Sommer, C. Gehring, M. Hoepflinger, M. Bloesch, and R. Siegwart, "Quadrupedal locomotion using hierarchical operational space control," *The International Journal of Robotics Research*, vol. 33, no. 8, pp. 1047–1062, 2014.

[33] L. Saab, O. E. Ramos, F. Keith, N. Mansard, P. Soueres, and J.-Y. Fourquet, "Dynamic whole-body motion generation under rigid contacts and other unilateral constraints," *IEEE Transactions on Robotics*, vol. 29, no. 2, pp. 346–362, 2013.

[34] P. M. Wensing and D. E. Orin, "Generation of dynamic humanoid behaviors through task-space control with conic optimization," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, IEEE, 2013, pp. 3103–3109.

[35] Y. Nakamura and H. Hanafusa, "Inverse kinematic solutions with singularity robustness for robot manipulator control," *ASME, Transactions, Journal of Dynamic Systems, Measurement, and Control*, vol. 108, pp. 163–171, 1986.

[36] C. W. Wampler, "Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 16, no. 1, pp. 93–101, 1986.

[37] M. Mistry, J. Nakanishi, G. Cheng, and S. Schaal, "Inverse kinematics with floating base and constraints for full body humanoid robot control," in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*, IEEE, 2008, pp. 22–27.

[38] F. Grasser, A. D'arrigo, S. Colombi, and A. C. Rufer, "Joe: A mobile, inverted pendulum," *IEEE Transactions on industrial electronics*, vol. 49, no. 1, pp. 107–114, 2002.

[39] J. Li, X. Gao, Q. Huang, Q. Du, and X. Duan, "Mechanical design and dynamic modeling of a two-wheeled inverted pendulum mobile robot," in *2007 IEEE International Conference on Automation and Logistics*, IEEE, 2007, pp. 1614–1619.

[40] V. Coelho, S. Liew, K. Stol, and G. Liu, "Development of a mobile two-wheel balancing platform for autonomous applications," in *Mechatronics and Machine Vision in Practice, 2008. M2VIP 2008. 15th International Conference on*, IEEE, 2008, pp. 575–580.

[41] K. Pathak, J. Franch, and S. K. Agrawal, "Velocity and position control of a wheeled inverted pendulum by partial feedback linearization," *Robotics, IEEE Transactions on*, vol. 21, no. 3, pp. 505–513, 2005.

[42] D. Angeli, "Almost global stabilization of the inverted pendulum via continuous state feedback," *Automatica*, vol. 37, no. 7, pp. 1103–1108, 2001.

[43] H. Marzi, "Multi-input fuzzy control of an inverted pendulum using an armature controlled dc motor," *Robotica*, vol. 23, no. 06, pp. 785–788, 2005.

[44] W. Qingcheng and F. Jian, "Fuzzy immune pd algorithm applied in the self-balancing two-wheeled robot," in *Future Generation Communication and Networking (FGCN), 2014 8th International Conference on*, IEEE, 2014, pp. 112–115.

[45] T.-J. Ren, T.-C. Chen, and C.-J. Chen, "Motion control for a two-wheeled vehicle using a self-tuning pid controller," *Control Engineering Practice*, vol. 16, no. 3, pp. 365–375, 2008.

[46] S. Jung and S. S. Kim, "Control experiment of a wheel-driven mobile inverted pendulum using neural network," *IEEE Transactions on Control Systems Technology*, vol. 16, no. 2, pp. 297–303, 2008.

[47] J. Wu and S. Jia, "Ts adaptive neural network fuzzy control applied in two-wheeled self-balancing robot," in *Strategic Technology (IFOST), 2011 6th International Forum on*, IEEE, vol. 2, 2011, pp. 1023–1026.

[48] J. Wu and W. Zhang, "Design of fuzzy logic controller for two-wheeled self-balancing robot," in *Strategic Technology (IFOST), 2011 6th International Forum on*, IEEE, vol. 2, 2011, pp. 1266–1270.

[49] L. B. Prasad, B. Tyagi, and H. O. Gupta, "Optimal control of nonlinear inverted pendulum dynamical system with disturbance input using pid controller & lqr," in *Control System, Computing and Engineering (ICCSCE), 2011 IEEE International Conference on*, IEEE, 2011, pp. 540–545.

[50] S. Ahmad, M. O. Tokhi, and S. F. Toha, "Genetic algorithm optimisation for fuzzy control of wheelchair lifting and balancing," in *Computer Modeling and Simulation, 2009. EMS'09. Third UKSim European Symposium on*, IEEE, 2009, pp. 97–101.

[51] K. M. GOHER and M. Tokhi, "Ga-optimised steering and position control of a two-wheeled vehicle with an extended rod–a simulation study," in *The 12th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines, Istanbul, Turkey*, 2009, pp. 66–74.

[52] K. M. Goher and M. Tokhi, "Genetic algorithm based modeling and control of a two wheeled vehicle with an extended rod, a lagrangian based dynamic approach," in *Cybernetic Intelligent Systems (CIS), 2010 IEEE 9th International Conference on*, IEEE, 2010, pp. 1–6.

[53] Z.-Q. Guo, J.-X. Xu, and T. H. Lee, "Design and implementation of a new sliding mode controller on an underactuated wheeled inverted pendulum," *Journal of the Franklin Institute*, vol. 351, no. 4, pp. 2261–2282, 2014.

[54] J. Akesson, A. Blomdell, and R. Braun, "Design and control of yaipan inverted pendulum on two wheels robot," in *2006 IEEE Conference on Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control*, IEEE, 2006, pp. 2178–2183.

[55] Y. Ha and S. Yuta, "Trajectory tracking control for navigation of self-contained mobile inverse pendulum," in *Intelligent Robots and Systems' 94.'Advanced Robotic Systems and the Real World', IROS'94. Proceedings of the IEEE/RSJ/GI International Conference on*, IEEE, vol. 3, 1994, pp. 1875–1882.

[56] H. Jian-hai, Z. Shu-shang, L. Ji-shun, and L. Hang, "Research on developed parallel two-wheeled robot and its control system," in *2008 IEEE International Conference on Automation and Logistics*, IEEE, 2008, pp. 2471–2475.

[57] Y. Kim, S.-H. Lee, and D. H. Kim, "Dynamic equations of a wheeled inverted pendulum with changing its center of gravity," in *Control, Automation and Systems (ICCAS), 2011 11th International Conference on*, IEEE, 2011, pp. 853–854.

[58] J. Lien, L. Tu, W. Ross, and C. Burvill, "Implementation issues for an inexpensive inverted-pendulum mobile robot," in *2006 International Conference on Information and Automation*, IEEE, 2006, pp. 372–377.

[59] T. Nomura, Y. Kitsuka, H. Suemitsu, and T. Matsuo, "Adaptive backstepping control for a two-wheeled autonomous robot," in *ICCAS-SICE, 2009*, IEEE, 2009, pp. 4687–4692.

[60] R. C. Ooi, "Balancing a two-wheeled autonomous robot," *University of Western Australia*, vol. 3, 2003.

[61] J.-S. Hu and M.-C. Tsai, "Design of robust stabilization and fault diagnosis for an auto-balancing two-wheeled cart," *Advanced Robotics*, vol. 22, no. 2-3, pp. 319–338, 2008.

[62] T. Takei, R. Imamura, *et al.*, "Baggage transportation and navigation by a wheeled inverted pendulum mobile robot," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 10, pp. 3985–3994, 2009.

[63] M.-C. Tsai and J.-S. Hu, "Pilot control of an auto-balancing two-wheeled cart," *Advanced Robotics*, vol. 21, no. 7, pp. 817–827, 2007.

[64] J. Wu, Y. Liang, and Z. Wang, "A robust control method of two-wheeled self-balancing robot," in *Strategic Technology (IFOST), 2011 6th International Forum on*, IEEE, vol. 2, 2011, pp. 1031–1035.

[65] T. R. Kane and D. A. Levinson, "The use of kane's dynamical equations in robotics," *The International Journal of Robotics Research*, vol. 2, no. 3, pp. 3–21, 1983.

[66] D. S. Bernstein, "Matrix mathematics: Theory," *Facts, and Formulas with*, 2005.

[67] K. S. Miller, "On the inverse of the sum of matrices," *Mathematics Magazine*, vol. 54, no. 2, pp. 67–72, 1981.

[68] M. Zafar, *Simulation of whole-body control of a wheeled inverted pendulum humanoid*, `https://youtu.be/4pNd84xLLUA`, [Online; accessed 28-July-2016], 2016.

[69] D. Jacobson and D. Mayne, *Differential dynamic programming*, ser. Modern analytic and computational methods in science and mathematics. American Elsevier Pub. Co., 1970.

[70] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, IEEE, 2012, pp. 4906–4913.

[71] N. Dantam and M. Stilman, "Robust and efficient communication for real-time multi-process robot software," in *Humanoid Robots (Humanoids), 2012 12th IEEE-RAS International Conference on*, IEEE, 2012, pp. 316–322.

[72] M. Zafar, S. Hutchinson, and E. A. Theodorou, "Hierarchical optimization for whole-body control of wheeled inverted pendulum humanoids," *arXiv preprint arXiv:1810.03074*, 2018.

[73] A. A. Bature, S. Buyamin, M. N. Ahmad, and M. Muhammad, *A comparison of controllers for balancing two wheeled inverted pendulum robot*, 2014.

[74] M. Zafar and H. I. Christensen, "Whole body control of a wheeled inverted pendulum humanoid," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, 2016, pp. 89–94.

[75] L. Canete and T. Takahashi, "Disturbance compensation in pushing, pulling, and lifting for load transporting control of a wheeled inverted pendulum type assistant robot using the extended state observer," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5373–5380.

[76] J. Carpentier, M. Benallegue, N. Mansard, and J. P. Laumond, "Center-of-mass estimation for a polyarticulated system in contact: A spectral approach," *IEEE Transactions on Robotics*, vol. 32, no. 4, pp. 810–822, 2016.

[77] G. G. Muscolo, C. T. Recchiuto, C. Laschi, P. Dario, K. Hashimoto, and A. Takanishi, "A method for the calculation of the effective center of mass of humanoid robots," in *2011 11th IEEE-RAS International Conference on Humanoid Robots*, 2011, pp. 371–376.

[78] M. Kudruss, M. Naveau, O. Stasse, N. Mansard, C. Kirches, P. Soueres, and K. Mombaur, "Optimal control for whole-body motion generation using center-of-mass dynamics for predefined multi-contact configurations," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, 2015, pp. 684–689.

[79] S. Kwon and Y. Oh, "Estimation of the center of mass of humanoid robot," in *2007 International Conference on Control, Automation and Systems*, 2007, pp. 2705–2709.

[80] E. Sihite and T. Bewley, "Attitude estimation of a high-yaw-rate mobile inverted pendulum; comparison of extended kalman filtering, complementary filtering, and motion capture," in *2018 Annual American Control Conference (ACC)*, 2018, pp. 5831–5836.

[81] A. Pajon, S. Caron, G. D. Magistri, S. Miossec, and A. Kheddar, "Walking on gravel with soft soles using linear inverted pendulum tracking and reaction force distribution," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, 2017, pp. 432–437.

[82] Z. Gao, "Active disturbance rejection control: A paradigm shift in feedback control system design," in *2006 American Control Conference*, 2006.

[83] D. Luo, Y. Wang, and X. Wu, "Online learning of com trajectory for humanoid robot locomotion," in *2012 IEEE International Conference on Mechatronics and Automation*, 2012, pp. 1996–2001.

[84] Q. Chen, H. Cheng, C. Yue, R. Huang, and H. Guo, "Step length adaptation for walking assistance," in *2017 IEEE International Conference on Mechatronics and Automation (ICMA)*, 2017, pp. 644–650.

[85] L. Yang, Z. Liu, and Y. Zhang, "Dynamic balance control of biped robot using optimized slfns," in *2016 Chinese Control and Decision Conference (CCDC)*, 2016, pp. 5303–5307.

[86] T. Kim and H. J. Kim, "Path tracking control and identification of tire parameters using on-line model-based reinforcement learning," in *2016 16th International Conference on Control, Automation and Systems (ICCAS)*, 2016, pp. 215–219.

[87] L. Jamone, B. Damas, and J. Santos-Victor, "Incremental learning of context-dependent dynamic internal models for robot control," in *2014 IEEE International Symposium on Intelligent Control (ISIC)*, 2014, pp. 1336–1341.

[88] L. Jiang, H. Qiu, Z. Wu, and J. He, "Active disturbance rejection control based on adaptive differential evolution for two-wheeled self-balancing robot," in *2016 Chinese Control and Decision Conference (CCDC)*, 2016, pp. 6761–6766.

[89] X. Ruan, X. Wang, X. Zhu, Z. Chen, and R. Sun, "Active disturbance rejection control of single wheel robot," in *Proceeding of the 11th World Congress on Intelligent Control and Automation*, 2014, pp. 4105–4110.

[90] D. Wei, C. Ren, M. Zhang, X. Li, S. Ma, and C. Mu, "Position/force control of a holonomic-constrained mobile manipulator based on active disturbance rejection control," in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017, pp. 6751–6756.

[91] R. Miklosovic and Z. Gao, "A dynamic decoupling method for controlling high performance turbofan engines," in *Proc. of the 16th IFAC World Congress*, Czech Republic, vol. 16, 2005, pp. 482–488.

[92] Georgia Tech Graphics Lab and Humanoid Robotics Lab, *Dart (dynamic animation and robotics toolkit)*, `http://dartsim.github.io/index.html`.

[93] M. Stilman, J. Olson, and W. Gloss, "Golem krang: Dynamically stable humanoid robot for mobile manipulation," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 3304–3309.

[94] A. A. Bature, S. Buyamin, M. N. Ahmad, and M. Muhammad, "A comparison of controllers for balancing two wheeled inverted pendulum robot," *International Journal of Mechanical & Mechatronics Engineering*, vol. 14, no. 3, pp. 62–68, 2014.

[95] A. Khosla, G Leena, and M. Soni, "Performance evaluation of various control techniques for inverted pendulum," *Performance Evaluation*, vol. 3, no. 4, pp. 1096–1102, 2013.

[96] D. Nguyen-Tuong and J. Peters, "Using model knowledge for learning inverse dynamics.," in *ICRA*, 2010, pp. 2677–2682.

[97] Y. Pan and E. Theodorou, "Probabilistic differential dynamic programming," in *Advances in Neural Information Processing Systems*, 2014, pp. 1907–1915.

[98] Y. Pan and X. Yan, "Prediction under uncertainty in sparse spectrum gaussian processes with applications to filtering and control," in *Proceedings of the 34th International Conference on Machine Learning*, vol. 70, 2017.

[99] L. Wang, E. A. Theodorou, and M. Egerstedt, "Safe learning of quadrotor dynamics using barrier certificates," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2018, pp. 2460–2465.